OCCASION

This publication has been made available to the public on the occasion of the 50th anniversary of the United Nations Industrial Development Organisation.

DISCLAIMER

FAIR USE POLICY

CONTACT

Please contact publications@unido.org for further information concerning UNIDO publications.

For more information about UNIDO, please visit us at www.unido.org

UNITED NATIONS INDUSTRIAL DEVELOPMENT ORGANIZATION
Vienna International Centre, P.O. Box 300, 1400 Vienna, Austria
Tel: (+43-1) 26026-0 · www.unido.org · unido@unido.org

RESTRICTED

19233

DP/ID/SER.A/1518
28 October 1991
ORIGINAL: ENGLISH

INTEGRATED INFORMATION NETWORK FOR EFFECTIVE MANAGEMENT OF RESEARCH AND DEVELOPMENT INSTITUTIONAL ACTIVITIES

DP/EGY/88/031

THE ARAB REPUBLIC OF EGYPT

Technical report: Integrated information network*

Prepared for the Government of the Arab Republic of Egypt
by the United Nations Industrial Development Organization,
acting as executing agency for the United Nations Development Programme

Based on the work of Mieczyslaw Muraszkiewicz,
chief technical adviser

Backstopping officers: M. Boutoussov and J. Pavlik,
Institutional Infrastructure Branch

United Nations Industrial Development Organization
Vienna

* This document has not been edited.

V.91 30096

TABLE OF CONTENTS

Explanatory Notes

Value of the local currency "Polish zloty" - zl
1 US $ = 11,120 zl (Oct. 1991)

Abbreviations

CTA	Chief Technical Adviser
RAMSES	Research-Access data Management System for Engineers and Scientists
R+D	Research and Development

Abstract

Muraszkiewicz, M., Chief Technical Adviser (CTA)
UNIDO

Integrated Information Network for Effective Management of R+D Institutional Activities
Report /EGY/88/031

This report presents the results of the work of the CTA over the period from 15 Aug. through 30. Sept. 1991 in Warsaw within the project: "Integrated Information Network for Effective Management of R+D Institutional Activities (DP/EGY/88/031). The work was aimed at: (i) to test thoroughly the existing RAMSES software prototype; (ii) to refine and extend the RAMSES software; (iii) to prepare the tools for transferring the RAMSES software from the PC level to the HP 3000 level; (iv) to draft the guidelines on the usage of the RAMSES database, including data inputting.

INTRODUCTION

The Chief Technical Adviser (CTA) worked in Warsaw to fulfill the recommendation being the result of the Project Revision Proposal. The assignment lasted over the period of Aug. 15 through Sept.30, 1991. The continuation of the present mission is planned for the mid November, 1991 in order to complete the project.

The main immediate objective of the project "Integrated Information Network for Effective Management of R+D Institutional Activities (DP/EGY/88/031) is the following: to design and establish a nucleus of the integrated information network (later on called RAMSES Research-Access data Management System for Engineers and Scientists) in order to provide facilities for efficient management and co-ordination of R+D institutions belonging to the research infrastructure of the Ministry of Industry in Egypt.

Briefly, so far the Ministry has obtained as a result of joint effort of the Ministry personnel and UNIDO consultants:

(i) detailed concept of the management system on R+D activities based on broad analysis of actual needs and existing facilities;

(ii) computer hardware composed mainly of the HP VECTRA QS/165 microcomputers, the HP 3000 minicomputer and telecommunication equipment to ensure data transmission on the basis of X.25 international standard protocol;

(iii) prototype of database software (shell) implemented on the HP VECTRA QS/165 microcomputer under Micro CDS ISIS software;

(iv) training of 2 persons as database administrators and application programmers;

(v) extensive training of three nationals in the field of designing and and implementing advanced computerized industrial information systems.

CONCLUSIONS AND RECOMMENDATIONS

1. Conclusions and recommendations local to the work specified by the Job description are given in Section III - Action Programme (Follow-up).

2. Other conclusions and recommendations are the same as provided in the CTA's report DP/ID/SER.A/1483 of June 27. 1991.

I. ACTIVITIES

The following activities according to the Job Description were undertaken.

1. To test thoroughly the RAMSES software prototype on the actual data sample collected from R & D organizations.

2. To refine and extend the following functions of the RAMSES software: (i) sorting and printing; (ii) data editing by means of look up tables; (iii) customization of database administrator options; (iv) retrieval by newly defined fields.

3. To prepare the software tools for transferring the RAMSES software from the PC level (under the MICRO CDS ISIS shell) to the HP 3000 level (under the MINISIS on-line package).

4. To draft the guidelines on the usage of the RAMSES database, including data inputting. The text should be later on (during the assignment in Cairo) discussed with the representatives of terminal users and refined.

II. OUTPUTS

All the tasks specified in Section I were completed and the objectives achieved. They are attached to this report in the following form:

(i) A report containing PASCAL programmes entitled "CDS ISIS PASCAL Programs for the RAMSES Database";
(ii) A report "Field Descriptions of the RAMSES Database" which is a draft where the guidelines on the data inputting are given;
(iii) A diskette containing the developed software.

III. ACTION PROGRAMME (FOLLOW-UP)

Below are given the most important tasks to be undertaken in order to accomplish the project.

to the Ministry of Industry:

1. To collect detailed comments and remarks from the RAMSES network users on the prototype software and extended data input forms.

2. To collect updated data on R+D activities from the participants of the network.

3. To acquire the MINISIS package and to train 2-3 persons on its usage (through the Arab League Documentation Center based in Cairo which does it free of charge).

4. To acquire 2 public ("normal") telephone lines in ARENTO (national telephone service company) and a leased line in EGYPTNET (provider of X.25 service in Egypt).

5. To establish all network connections and test them (according to the specification by UNIDO consultant Mr. W. Bauerfeld), including purchase and installation of X.25 component for the ICL ME 29 computer.

   This should be done by ORASCOM and CODEX companies under the Ministry's supervision.

6. To formulate highlights of the national R&D policy for the 3-5 coming years.

to CTA:

1. To implement the results of the present work at the Ministry of industry.

2. To implement remaining tasks, specified in the Job Description, in order to complete the project.

ANNEX I

FIELD DESCRIPTIONS OF THE RAMSES DATABASE

0. INTRODUCTION

This document contains the descriptions of the RAMSES (Research data Access Management System for Engineers and Scientists) data fields concerned with the object types ENTERPRISE and PROJECT. In terms of the MICRO-ISIS the fields are also defined in the FDT table. The order of fields descriptions corresponds to their order in the FDT table. All the fields are presented in a standard way providing the parameters of a field and its characteristic.

The RAMSES database is implemented for IBM PC XT/AT and compatible microcomputers. The software is based on the Micro CDS/ISIS, ver. 2.3, package developed by UNESCO.

1. Field descriptions

Field name: *Record type*
Tag: 003
Type (A - alpha, N - numerics, X - A + N, P - pattern): A
Subfields: None
Repeatability: Non-repeatable
Description: The field specifies record type in heterogeneous databases. The following record types are possible:
- ENTERPRISE
- PROJECT

Associated table(s): None
Example(s): E

Field name: *Archives/Current*
Tag:
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None
Repeatability: Non-repeatable
Description:
Associated table(s): None
Example(s):

Field name: *Financial Year*
Tag: 5
Type (A - alpha, N - numerics, X - A + N, P - pattern): P - 99999
Subfields: None
Repeatability: Non-repeatable
Description: The financial year is specified.
Associated table(s): None
Example(s): 1991

Field name: *Name of the institution*
Tag: 10
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: The field contains a full name, preferably in english, of the institution being described.
Associated table(s): None
Example: Institute for Scientific, Technical and Economic Information

Field name: *Acronym*
Tag: 15
Type (A - alpha, N - numerics, X - A + N, P - pattern): X

Subfields: None
Repeatability: Non-repeatable
Description: If the acronym does not exist it should be coined from the first letters of the meaningful words in the English name of the enterprise.
Associated table(s): None
Example(s): for the example above - ISTEI

Field name: *Institution code*
Tag: 17
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: For each participating institution a unique code is to be assigned by the RAMSES staff. A method of coding must established. The codes are to be used for linking records of different types.
Associated table(s): None
Example(s): ISTEI-345

Field name: *Supervising Organization*
Tag: 18
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: A full name of the supervising organization.
Associated table(s): None
Example(s): Ministry of Industry.

Field name: *Location*
Tag: 19
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: The field contairs the address of the headquarters.
Associated table(s): None
Example(s): P.O.Box 187, 2 King Road, Cairo 7, Egypt

Field name: *Director/Head*
Tag: 25
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: The field contains the name, surnames and/or initials, title.
Associated table(s): None

Example(s): Prof. De Bocca J.I., Jr.

Field name: *Contact persons*
Tag: 26
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields:
Repeatability: Non-repeatable
Description: The field contains name(s) of a contact person(s).
Associated table(s): None
Example(s): Eng. Nagwa; Eng. Shams

Field name: *Contact person address*
Tag: 20
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: The address of a contact person.
Associated table(s): None
Example(s): P.O.Box 187, 2 King Road, Cairo 7, Egypt

Field name: *Phone*
Tag: 21
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: The field contains phone numbers to the institution.
Associated table(s): None
Example(s): (809)984-2354, (809)984-2345

Field name: *Tlx*
Tag: 22
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: The field contains the telex number(s); the format of the field is free.
Associated table(s): None
Example(s): 12346 FPI

Field name: *Fax*
Tag: 23
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None
Repeatability: Non-repeatable

Description: The fax number is given in the form of the phone number.
Associated table(s): None
Example(s): 12-23-578, 34-45-621

Field name: *Electronic Mail*
Tag: 24
Type (A = alpha, N = numerics, X = A + N, P = pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: The field contains E-mail address.
Associated table(s): None
Example(s):EMU@INT-1

Field name: *ISIC*
Tag: 30
Type (A = alpha, N = numerics, X = A + N, P = pattern): X
Subfields: None
Repeatability: Repeatable
Description: ISIC codes taken from Thesaurus of Industrial Development Terms. If more than one code is to be selected each code is in the separate field occurrencei.e. the codes are separated by means of %
Associated table(s): None
Example(s): 1345%4128%3321

Field name: *Sector*
Tag: 35
Type (A = alpha, N = numerics, X = A + N, P = pattern): X
Subfields: None
Repeatability: Repeatable
Description: Sectors according to the choosen (e.g. UNIDO) classification have to be provided. The codes are to be separated by %.
Associated table(s): None
Example(s): light industry%textile

Field name: *Sub-sector*
Tag: 40
Type (A = alpha, N = numerics, X = A + N, P = pattern): X
Subfields: None
Repeatability: Repeatable
Description: Sub- sectors according to the choosen (e.g. UNIDO) classification have to be provided. The codes are to be separated by %.
Associated table(s): None
Example(s): textile%shirts

Field name: *Type of enterprise*
Tag: 50, 51, 52
Type (A = alpha, N = numerics, X = A + N, P = pattern): A
Subfields: None
Repeatability: Non-repeatable
Description: Three types of enterprise are allowed: Public (50), Private (51) and mixte (52). Put Y in appropriate field if the enterprise is relevant to the name of the field; otherwise put N.
Associated table(s): None
Example(s): Y, N, N

Field name: *Subsidiaries*
Tag: 66
Type (A = alpha, N = numerics, X = A + N, P = pattern): X
Subfields: None
Repeatability: Repeatable
Description: List of all the subsidiaries of the enterprise. The items are separeted by %.
Associated table(s): None
Example(s):

Field name: *Total Capital Invest. in EP*
Tag: 70
Type (A = alpha, N = numerics, X = A + N, P = pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: The field contains the information on the institution total capital investment in E.P. - free text
Associated table(s): None
Example(s): EP 1,230,000

Field name: *Total Capital Invest. in* US$
Tag: 70
Type (A = alpha, N = numerics, X = A + N, P = pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: The field contains the information on the institution total capital investment in US$. - free text
Associated table(s): None
Example(s): US$ 230,000

Field name: *Annual Operating Expenses EP*
Tag: 72
Type (A = alpha, N = numerics, X = A + N, P = pattern): X

Subfields: None
Repeatability: Non-repeatable
Description: The field contains the informaticn on the institution Annual Operating Expenses in E.P.
Associated table(s): None
Example(s): E.P. 200,000,000

Field name: *Annual Operating Expenses* US$
Tag: 73
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: The field contains the information on the institution Annual Operating Expenses in US$.
Associated table(s): None
Example(s): US$ 200,000

Field name: *Annual Revenue and Fees EP*
Tag: 74
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: The field contains the information on the institution Annual Revenue and Fees in E.P.
Associated table(s): None
Example(s): E.P. 7, 234,400

Field name: *Annual Revenue and Fees* US$
Tag: 75
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: The field contains the information on the institution Annual Revenue and Fees in US$.
Associated table(s): None
Example(s): US$ 235,400

Field name: Total Annual Saleries EP
Tag: 76
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: The field contains the information on the institution Total Annual Saleries in E.P.
Associated table(s): None
Example(s): E.P. 2,341,000

Field name: *Number of Professionals*
Tag: 80
Type (A = alpha, N = numerics, X = A + N, P = pattern): N
Subfields: None
Repeatability: Non-repeatable
Description: The number of professionals employed by the enterprise is given.
Associated table(s): None
Example(s): 34

Field name: *Other*
Tag: 82
Type (A = alpha, N = numerics, X = A + N, P = pattern): N
Subfields: None
Repeatability: Non-repeatable
Description: The number of non-professionals employed by the enterprise is given.
Associated table(s): None
Example(s): 45

Field name: *Pure Research*
Tag: 90
Type (A = alpha, N = numerics, X = A + N, P = pattern): N
Subfields: None
Repeatability: Non-repeatable
Description: The number of pure research projects carried out by the enterprise is given.
Associated table(s): None
Example(s): 2

Field name: *Applied research*
Tag: 91
Type (A = alpha, N = numerics, X = A + N, P = pattern): N
Subfields: None
Repeatability: Non-repeatable
Description: The number of pure research projects carried out by the enterprise is given.
Associated table(s): None
Example(s): 2

Field name: *Training*
Tag: 92

Type (A = alpha, N = numerics, X = A + N, P = pattern): N
Subfields: None
Repeatability: Non-repeatable
Description: The number of training courses carried out by the enterprise is given.
Associated table(s): None
Example(s): 3

Field name: *Consultations*
Tag: 93
Type (A = alpha, N = numerics, X = A + N, P = pattern): N
Subfields: None
Repeatability: Non-repeatable
Description: The number of consultations carried out by the enterprise is given.
Associated table(s): None
Example(s): 2

Field name: *Services*
Tag: 94
Type (A = alpha, N = numerics, X = A + N, P = pattern): N
Subfields: None
Repeatability: Non-repeatable
Description: The number of services rendered by the enterprise is given.
Associated table(s): None
Example(s): 2

Field name: *Installations*
Tag: 95
Type (A = alpha, N = numerics, X = A + N, P = pattern): N
Subfields: None
Repeatability: Non-repeatable
Description: The number of installations implemented by the enterprise is given.
Associated table(s): None
Example(s): 2

Field name: *Maintenence*
Tag: 96
Type (A = alpha, N = numerics, X = A + N, P = pattern): N
Subfields: None
Repeatability: Non-repeatable
Description: The number of maintenence rendered by the enterprise is given.
Associated table(s): None

Example(s): 2

Field name: *List of Main Departments*
Tag: 100
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None
Repeatability: Repeatable
Description: Give a list of the main departments of the enterprise. The names must be separated by %.
Associated table(s): None
Example(s): Research Dept.%Microcomputer Laboratory%Bookkeeping

Field name: *List of Resources Utilised*
Tag: 105
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: Inup
^I - place of use
^n - name of resource
^u - level of utilisation
^p - price
Repeatability: Repeatable
Description: List of main resources PIs, note the subfields. Items are separated by %.
Associated table(s): None
Example(s): ^IMicrolab^nVECTRA 1500 PC^u300 hrs/yr^pUS$ 3,500

Field name: *List of Resources Required*
Tag: 106
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: Inup
^I - place of use
^n - name of resource
^u - level of utilisation
^p - price
Repeatability: Repeatable
Description: List of main resources PIs, note the subfields. Items are separated by %.
Associated table(s): None
Example(s): ^IMicrolab^nSUN PC^u100 hrs/yr^pUS$ 23,500

Field name: *List of Projects (IDs)*
Tag: 107
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None
Repeatability: Repeatable
Description: List pf project codes carried out by the enterprise.
Associated table(s): None

Example(s): PR-7/91%PR-A/91%SE-QR/91

Field name: *Brief Statement of Area of Specialization*
Tag: 110
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: Short information on the activities carried out by the enterprise.
Associated table(s): None
Example(s): The Institute is carring out research on industrial information systems, in particular on CD-ROM databases.

Field name: *Principal references*
Tag: 500
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: Names (and addresses, if possible) of the enterprises and institutions which could provide references - free text.
Associated table(s): None
Example(s): Institute of Food Processing; Textile Institute.

Field name: *Publications*
Tag: 520
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: List of main publications issued and or periodicals edited.
Associated table(s): None
Example(s): News on textile products

Field name: *Exportation*
Tag: 525
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: Information on export products provided by the enterprise - free text.
Associated table(s): None
Example(s): 2400 b/s modems; RS 232 interfaces.

Field name: *Project Title*
Tag: 200

Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: Full title of the project
Associated table(s): None
Example(s): Development of the classification system for shoes.

Field name: *Code of the project*
Tag: 210
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: Unique code of the project. To be used as a reference from the object type ENTERPRISE.
Associated table(s): None
Example(s): PR-7/A/91

Field name: *Status of the project*
Tag: 215-9
Type (A - alpha, N - numerics, Y - A + N, P - pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: Status of the project:
Planned 215
Approved 216
Initiated 217
Completed 218
Canceled 219
Put Y if the actual status of the project correspond to the field name; otherwise put N.
Associated table(s): None
Example(s): Y,N,N,N,N

Field name: *Type of the project*
Tag: 220, 222, 224, 226, 228, 230,232
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: Type of the project
Research 220
Development 222
Training 224
Consultations 226
Service 228
Installation 230
Maintenance 232
Put Y if the actual type of the project correspond to the field name; otherwise put N.

Associated table(s): None
Example(s): Y,Y,N,N,N,N,N

Field name: *Department*
Tag: 235
Type (A = alpha, N = numerics, X = A + N, P = pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: Write the name(s) of the department of the enterprise where the project is executed.
Associated table(s): None
Example(s): Microcomputer Laboratory

Field name: *Project Leader*
Tag: 240
Type (A = alpha, N = numerics, X = A + N, P = pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: Write the name and title, if any, of the project leader.
Associated table(s): None
Example(s): Dr. Yousuf Redda.

Field name: *Summary of Objectives*
Tag: 250
Type (A = alpha, N = numerics, X = A + N, P = pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: Summarize shortly the main objectives of the project.
Associated table(s): None
Example(s): The main goal of the project is to establish an information system on the exported products manufactured by the Institute.

Field name: *Contributing Organization(s)*
Tag: 260
Type (A = alpha, N = numerics, X = A + N, P = pattern): X
Subfields: None
Repeatability: Repeatable
Description: Give the names of the contributing (financialy and in kind) organizations. The items have to be separated by %.
Associated table(s): None
Example(s): IFP%ASTIC

Field name: *Client Organization(s)*
Tag: 270

Type (A = alpha, N = numerics, X = A + N, P = pattern): X
Subfields: None
Repeatability: Repeatable
Description: Give the names of the clientorganizations. The items have to be separated by %.
Associated table(s): None
Example(s): Ministry of Industry%ASTIC

Field name: *Starting Date of the project*
Tag: 280
Type (A = alpha, N = numerics, X = A + N, P = pattern): P 99-99-99
Subfields: None
Repeatability: Non-repeatable
Description: The field contains the date of starting the project. The field is entered in the way YY-MM-DD, where:
YY are the last two digits of the year;
MM are the digits for the month;
DD are the digits for the day.
Associated table(s): None
Example(s): 92-02-25

Field name: *Completion Date*
Tag: 282
Type (A = alpha, N = numerics, X = A + N, P = pattern): P 99-99-99
Subfields: None
Repeatability: Non-repeatable
Description: The field contains the date of finishing the project. The field is entered in the way YY-MM-DD, where:
YY are the last two digits of the year;
MM are the digits for the month;
DD are the digits for the day.
Associated table(s): None
Example(s): 92-02-25

Field name: *Duration of the project*
Tag: 284
Type (A = alpha, N = numerics, X = A + N, P = pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: The duration of the project in months is to be written.
Associated table(s): None
Example(s): 24

Field name: *Next Review Date*
Tag: 286
Type (A = alpha, N = numerics, X = A + N, P = pattern): P 99-99-99
Subfields: None

Repeatability: Non-repeatable
Description: The field contains the date next revision of the project. The field is entered in the way YY-MM-DD, where:
YY are the last two digits of the year;
MM are the digits for the month;
DD are the digits for the day.
Associated table(s): None
Example(s): 92-02-25

Field name: *Remarks*
Tag: 530
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: Additional remarks
Associated table(s): None
Example(s): The project is to be a model for other similar enterprises.

Field name: *Input by*
Tag: 900
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None
Repeatability: Non-repeatable
Description: The field contains the name of a person, who keyed in the data from R+D to the RAMSES system.
Associated table(s): None
Example(s): Smith, J.

Field name: *Date of entering data*
Tag: 901
Type (A - alpha, N - numerics, X - A + N, P - pattern): P 99-99-99
Subfields: None
Repeatability: Non-repeatable
Description: The field contains the date of entering the data into the system. The field is entered in the way YY-MM-DD, where:

YY are the last two digits of the year;
MM are the digits for the month;
DD are the digits for the day.
Associated table(s): None
Example(s): 92-02-25

Field name: *Update*
Tag: 902
Type (A - alpha, N - numerics, X - A + N, P - pattern): X
Subfields: None

Repeatability: Non-repeatable
Description: The field contains the name of a person, who updated the data.
Associated table(s): None
Example(s): Smith, J.

Field name: *Date of updating data*
Tag: 903
Type (A = alpha, N = numerics, X = A + N, P = pattern):
P 99-99-99
Subfields: None
Repeatability: Non-repeatable
Description: The field contains the date of updating the data in the system. The field is entered in the way YY-MM-DD, where:
YY are the last two digits of the year;
MM are the digits for the month;
DD are the digits for the day.
Associated table(s): None
Example(s): 92-02-25

CDS ISIS PASCAL PROGRAMS FOR THE RAMSES DATABASE

0. INTRODUCTION

This document contains CDS ISIS PASCAL programs dealing with the RAMSES (Research data Access Management System for Engineers and Scientists) database. RAMSES has been designed to provide information on:

(i) R + D institutes and organisation carrying out their activities in Egypt;

(ii) Projects performed by these organisations;

(iii) Experts available within these organisations and co-operating with them;

(iv) Queries and relevant responses addresse to the R + D organisations;

(v) Reports and other documents related to the execution of projects;

(vi) Software available within the R + D community;

(vii) Library activities related to the R + D organisations and projects.

The RAMSES database is implemented for IBM PC XT/AT and compatible microcomputers. The software is based on the Micro CDS/ISIS, ver. 2.3, package [1] developed by UNESCO, 1989.

```
Program BIBFOR(f: string) [menu];

{     This program selects one of the BIBLO display formats          }
{     A moving arrow and highlighting are used as markers            }
{     The program is invoked from the menu EFGEN, option F           }

                    { **** by M.Muraszkiewicz, Sept 4, 1991 **** }

var     sop, lin, col. hi, wi, le            :real;
        str, fornm                           :string;

Function SELECT(stlin,stcol,high,wide,len :real;str :string): real;
    {  This function allows for selecting an option by means    }
    {  of an arrow and highlighting.                            }
    {    Input paramaters:  stlin, stcol, high  wide, len       }
    {    Output parameters: SELECT, str                         }
var sc, i, k :real;
begin
i:= 0; k:= 0; str:= '';
repeat
CURSOR(stlin+1+i,stcol+wide);
writeln('<----');                                  { draw  new arrow    }
ATTR(' ',0,26,80,1);                               { hide a cursor      }
sc:= KBDKEY(str);
if sc = 80 then i:= i+1;                           { down arrow was met }
if sc = 72 then i:= i-1;                           { up arrow was met   }
if i > high-3 then i:= 0;                          { skip to the top    }
if i < 0 then i:=high-3;                           { skip to the bottom }
if sc=72 or sc=80 then
   begin
   CLEARBOX(stlin+1+k,stcol+wide,1,5,0);           { remove old arrow   }
   CHATTR(0,stlin+1+k,stcol+1,len);                { remove cld hlight  }
   CHATTR(2,stlin+1+i,stcol+1,len);                { put new highlight  }
   end;
k:= i;
until sc=28 or str='x' or str='X';                 { ENTER or X was met }
SELECT:= k;
end;

Procedure INDEX_BOX(stlin,stcol,high,wide,len :real);
{  Draws a box and fills it out with indexes names  }
begin
BOX(stlin,stcol,high,wide,1);
ATTR(' ',2,stlin+1,stcol+1,len);                   { 2 - hlight  }
CURSOR(stlin+1,stcol+1); write('General Format');
CURSOR(stlin+2,stcol+1); write('Short Format');
CURSOR(stlin+3,stcol+1); write('Proofreading Format');
CURSOR(stlin+high+1,stcol+11); write('X - Exit');
end;

Procedure MESSAGE;
begin
CURSOR(3,27); write('Format BIBLO');
CURSOR(22,1);
write('Select your format by means of arrows  ');
```

```
CURSOR(23,1); write('Type ENTER to confir you choice');
CURSOR(23,1); write('Strike X to quit');
end;
                { -------- Body of Program -------- }
begin
CLEAR;                                          { clear screen }
MESSAGE;                                        { display msgs }
CURSOR(4,3);

   { ---------- Display and pick up a format name ---------- }
   lin:= 8; col:= 22; hi:= 5; wi:= 39; le:= wi-2;
   INDEX_BOX(lin,col,hi,wi,le); CURSOR(lin-1,col+1); {draw a box + opt.}
   sop:= SELECT(lin,col,hi,wi,le,str);              { pick up by <--- }
   if str <> 'x' and str <> 'X' then
      begin                                         { X - Exit not met }
      { --------- formats --------- }
      if sop = 0 then fornm:='BIBLO';
      if sop = 1 then fornm:='ISFOR';
      if sop = 2 then fornm:='PROOF';
      GETFMT('@':fornm);
      end;
   f:= ' ';                                          { return to menu }
end.
```

```
Program BIBIND(s: string) [menu];
{     This program selects one of the BIBLO Indexes for printing     }
{     A moving arrow and highlighting are used as markers            }
{     The program is invoked from the menu FXPRT, option S           }

                        { **** by M.Muraszkiewicz, Oct. 2, 1991 **** }

var     sop1, sop2, lin, col, hi, wi, le      :real;
        str, sheet                            :string;


Procedure DSPL(nrow,ncol: real; text: string);
begin
CURSOR(nrow,ncol); WRITE(text);
end;

Function SELECT(stlin,stcol,high,wide,len :real;str :string): real;
    {  This function allows for selecting an option by means    }
    {  of an arrow and highlighting.                            }
    {     Input paramaters:  stlin, stcol, high, wide, len      }
    {    Output parameters: SELECT, str                         }
var sc, i, k :real;
begin
i:= 0; k:= 0; str:= '';
repeat
CURSOR(stlin+1+i,stcol+wide);
writeln('<----');                                  { draw  new arrow    }
ATTR(' ',0,26,80,1);                               { hide a cursor      }
sc:= KBDKEY(str);
if sc = 80 then i:= i+1;                           { down arrow was met }
if sc = 72 then i:= i-1;                           { up arrow was met   }
if i > high-3 then i:= 0;                          { skip tc the top    }
if i < 0 then i:=high-3;                           { skip to the bottom }
if sc=72 or sc=80 then
   begin
   CLEARBOX(stlin+1+k,stcol+wide,1,5,0);           { remove old arrow   }
   CHATTR(0,stlin+1+k,stcol+1,len);                { remove old hlight  }
   CHATTR(2,stlin+1+i,stcol+1,len);                { put new highlight  }
   end;
k:= i;
until (sc=28) or (str='x') or (str='X');           { ENTER or X was met }
SELECT:= k;
end;

Procedure INDEX_BOX1(stlin,stcol,high,wide,len :real);
{  Draws a box and fills it out with 2 options  }
begin
BOX(stlin,stcol,high,wide,1);
ATTR(' ',2,stlin+1,stcol+1,len);                   { 2 - hlight  }
DSPL(stlin+1,stcol+1,'Hits');
DSPL(stlin+2,stcol+1,'Whole base');
DSPL(stlin+high+1,stcol+4,'X - Exit');
end;
```

```
Procedure INDEX_BOX2(stlin,stcol,high,wide,len :real);
{  Draws a box and fills it out with indexes names  }
begin
BOX(stlin,stcol,high,wide,1);
ATTR(' ',2,stlin+1,stcol+1,len);                    { 2 - hlight  }
DSPL(stlin+1,stcol+1,'List of References');
DSPL(stlin+2,stcol+1,'Index  of Authors');
DSPL(stlin+3,stcol+1,'Subjet Index');
DSPL(stlin+4,stcol+1,'Collective Author Index');
DSPL(stlin+high+1,stcol+4,'X - Exit');
end;

Procedure MESSAGE;
begin
DSPL(2,30,'Index BIBLO');
CURSOR(22,1);
write('Select your format by means of arrows  ');
CURSOR(23,1); write('Type ENTER to confir you choice');
CURSOR(23,1); write('Strike X to quit');
end;
                { -------- Body of Program -------- }
begin
CLEAR;                                              { clear screen }
MESSAGE;                                            { display msgs }
CURSOR(5,5);

       { --------- Hits or Whole database ? ----------- }
write('Do you want to create hits or whole database index?');
lin:= 9; col:= 29; hi:= 4; wi:= 17; le:= wi-2;
s:= ' ';                                            { return to print menu }
INDEX_BOX1(lin,col,hi,wi,le);                       { draw a box + options }
sop1:= SELECT(lin,col,hi,wi,le,str);                { pick up by <----     }

CLEAR;
if (str <> 'x') and (str <> 'X') then
   begin                                                 { X - Exit not met }
   { ---------- Display and pick up AGNAT Indexes ----------- }
   MESSAGE; lin:= 8; col:= 23; hi:= 6; wi:= 35; le:= wi-2;
   INDEX_BOX2(lin,col,hi,wi,le); CURSOR(lin-1,col+1); {draw a box + opt.}
   if sop1 = 0 then write('HITS');
   if sop1 = 1 then write('TOUTE LA BASE');
   sop2:= SELECT(lin,col,hi,wi,le,str);                 { pick up by <--- }
   if (str <> 'x') and (str <> 'X') then
      begin                                             { X - Exit not met }
      { ------------------ hits ----------------- }
      if sop1 = 0 and sop2 = 0 then sheet:='fyb1h';
      if sop1 = 0 and sop2 = 1 then sheet:='fyb2h';
      if sop1 = 0 and sop2 = 2 then sheet:='fyb3h';
      if sop1 = 0 and sop2 = 3 then sheet:='fyb4h';
      { ------------- whole database ------------ }
      if sop1 = 1 and sop2 = 0 then sheet:='fyb1b';
      if sop1 = 1 and sop2 = 1 then sheet:='fyb2b';
      if sop1 = 1 and sop2 = 2 then sheet:='fyb3b';
      if sop1 = 1 and sop2 = 3 then sheet:='fyb4b';
```

```
            AUTOTYPE(sheet);
            CLEARMSG; CURSOR(22,21); write(sheet);
            s:= '.S';                                  { return & call a worksheet }
            end;
         end;
      end.
```

```
Program BIBWSH(w: string) [menu];
{     This program selects a worksheet for the BIBLO database       }
{     A moving arrow and highlighting are used as markers           }
{     The program is invoked from the menu FXE1, option W           }

                    { **** by M.Muraszkiewicz, Sept. 28, 1991 **** }

var    sop, d, lin, col, hi, wi, le          :real;
       str, sheet                            :string;

Function SELECT(stlin,stcol,high,wide,len :real;str :string): real;
    {  This function allows for selecting an option by means   }
    {  of an arrow and highlighting.                           }
    {    Input paramaters:  stlin, stcol, high, wide, len      }
    {    Output parameters: SELECT, str                        }
var sc, i, k :real;
begin
i:= 0; k:= 0; str:= '';
repeat
CURSOR(stlin+1+i,stcol+wide);
writeln('<----');                                  { draw  new arrow    }
ATTR(' ',0,26,80,1);                               { hide a cursor      }
sc:= KBDKEY(str);
if sc = 80 then i:= i+1;                           { down arrow was met }
if sc = 72 then i:= i-1;                           { up arrow was met   }
if i > high-3 then i:= 0;                          { skip to the top    }
if i < 0 then i:=high-3;                           { skip to the bottom }
if (sc=72) or (sc=80) then
   begin
   CLEARBOX(stlin+1+k,stcol+wide,1,5,0);           { remove old arrow   }
   CHATTR(0,stlin+1+k,stcol+1,len);                { remove old hlight  }
   CHATTR(2,stlin+1+i,stcol+1,len);                { put new highlight  }
   end;
k:= i;
until (sc=28) or (str='x') or (str='X');           { ENTER or X was met }
SELECT:= k;
end;

Procedure INDEX_BOX(stlin,stcol,high,wide,len :real);
{  Draws a box and fills it out with worksheets names  }
begin
BOX(stlin,stcol,high,wide,1);
ATTR(' ',2,stlin+1,stcol+1,len);                   { 2 - hlight  }
CURSOR(stlin+1,stcol+1); write('Journal papers');
CURSOR(stlin+2,stcol+1); write('Book Papers');
CURSOR(stlin+3,stcol+1); write('Raports');
CURSOR(stlin+4,stcol+1); write('Journal');
CURSOR(stlin+high,stcol+7); write('X - Exit');
end;

Procedure MESSAGE;
begin
CURSOR(3,21); write('BIBLO Worksheets');
CURSOR(17,12); write('Generals');
```

```
CURSOR(18,12); write('Specific');
write('Select your format by means of arrows  ');
CURSOR(23,1); write('Type ENTER to confir you choice');
CURSOR(23,1); write('Strike X to quit');
end;
                { -------- Body of Program -------- }
begin
CLEAR;                                              { clear screen }
MESSAGE;                                            { display msgs }
CURSOR(4,3);

   { ---------- Display and pick up a worksheet ---------- }
   lin:= 5; col:= 22; hi:= 10; wi:= 36; le:= wi-2;
   INDEX_BOX(lin,col,hi,wi,le); CURSOR(lin-1,col+1); {draw a box + opt.}
   sop:= SELECT(lin,col,hi,wi,le,str);              { pick up by <--- }
   if (str <> 'x') and (str <> 'X') then
      begin                                         { X - Exit not met }
      { --------- worksheets --------- }
      if sop = 0 then sheet:='TAJF';
      if sop = 1 then sheet:='TAMF';
      if sop = 2 then sheet:='TRF';
      if sop = 3 then sheet:='TJF';
      if sop = 4 then sheet:='TMF';
      if sop = 5 then sheet:='TZF';
      if sop = 6 then sheet:='TZMF';
      if sop = 7 then sheet:='BIBLO';
      d:= WORKSHEET(sheet);
      end;
   w:= ' ';                                         { return to menu }
end.
```

```
Program COPY(q: string) [menu];
{-------------------------------------------------------------}
{     This program copies the specified PROJECT record        }
{     The program is invoked from the EXE1 menu option C      }
{-------------------------------------------------------------}

              { **** by M.Muraszkiewicz, Sept. 18, 1991 **** }

var  d, m_l, n_r                                :real;
     r_type, wks_name, sc                       :string;

Procedure MESSAGE(lin1,col1:real;text1:string;
                  lin2,col2:real;text2:string;
                  lin3,col3:real;text3:string;
                  halt:string);

{-----------------------------------------------------}
{         Displays the messages                       }
{-----------------------------------------------------}

var du   :string;

begin
cursor(lin1,col1);
writeln(text1);
cursor(lin2,col2);
writeln(text2);
cursor(lin3,col3);
write(text3);
if halt='halt' then du:= INKEY;

end; {procedure}

Procedure TRANSFER(r_num_old :real;wks_name :string);

{-------------------------------------------------------------}
{    Copies the record r_num_old to the new one.              }
{    The field numbs are taken from the wks_name worksheet.   }
{-------------------------------------------------------------}

var  tagnu                                      :array[1..250] of real;
     r_num_new, n_occ, f_num, n_field, i, d     :real;
     nf, indx, l, k, tot_n_fld, f_ex_tag, j     :real;
     f_cont, line1, line12, s_n_field, seq      :string;
     str, pre                                   :string;

begin
         l:=23; k:=57;
         ATTR(' ',4,1,k,13);
         MESSAGE(1,k+1,'Please wait',1,1,'',26,1,'','');

tot_n_fld:= 0;                          { total number of fields }
pre:= 'a';
j:= 0;
```

```
f_ex_tag:= FILEXIST(path('dbn',10):pre:wks_name);

WHILE (f_ex_tag=0) DO
begin
   ASSIGN('INP',path('dbn',10):pre:wks_name);

{ --- get the number of fields and their tags from the worksheet --- }
   FOR i:=1 TO 12 DO
     begin
     readln(INP,line12);
     if i=1 then line1:= line12;
     end;

   n_field:= VAL(SUBSTR(line1,1,3));  { number of fields on worksheet }

   FOR i:= 1 TO n_field DO
     begin
     seq:= SUBSTR(line12,(i-1)*6+1,6);
     tagnu[i+tot_n_fld]:= VAL(seq);    { field tags are in tagnu      }
     end;

   tot_n_fld:= tot_n_fld + n_field;

   j:= j+1;
   pre:= CHR(97+j);                        { produces prefix b,c,d,e,.... }
   f_ex_tag:= FILEXIST(path('dbn',10):pre:wks_name);
end; { while }

r_num_new:= NEWREC;                         { create the new record    }
UPDATE;

FOR i:= 1 TO tot_n_fld DO
  begin
  indx:= tagnu[i];
  if indx <> 0 then
     begin
     d:= RECORD(r_num_old);
     n_occ:= NOCC(indx);
     FOR l:= 1 TO n_occ DO
       begin
       d:= RECORD(r_num_old);
       f_num:= FIELDN(indx,l);
       f_cont:= FIELD(f_num);
          { writeln('tagnu ',tagnu[i],'   old:  ',f_cont);}
       d:= RECORD(r_num_new);
       nf:= NFIELDS;
       d:= FLDADD(indx,nf+1,f_cont);
       UPDATE;
       d:= RECORD(r_num_new);
          { f_num:= FIELDN(indx,1);
            f_cont:= FIELD(f_num);
            writeln('tagnu ',tagnu[i],'   new:  ',f_cont); str:= inkey;}
       end; { for }
     end { if }
```

```
  end;   { for}
end; { procedure }

Function MFN_NUMBER(r_type,scode:string):real;

{-------------------------------------------------------}
{   Asks for the number of the record to be transferred   }
{   returns 0 if X or x met or wrong record type          }
{-------------------------------------------------------}

var    m_l, max, d    :real;
       seq, typ       :string;
begin
m_l:=0;
scode:='';
CLEARBOX(22,1,2,80,0);
CURSOR(24,1); writeln('Type X [Enter] to leave without moving the record');
repeat
  CLEARBOX(7,1,1,80,0); CURSOR(7,7);
  write('Pls, specify MFN of the record to be moved: ');
  readln(seq); m_l:=val(seq);
  UC(seq);                                  { upper case }
  max:= MAXMFN;
  CLEARBOX(9,1,1,80,0); CURSOR(9,7);
  if (m_l>=max) then
     begin
     CLEARBOX(9,1,1,80,0); CURSOR(9,7);
     writeln('Sorry, your MFN is wrong !   Try again, pls');
     end;
until ( (m_l<max) AND (m_l>0) OR (seq='X'));
MFN_NUMBER:= m_l;

if seq<>'X' then
begin
  d:= RECORD(m_l);
  typ:= FIELD(FIELDN(3,1));   { get the record type }
  if typ<>r_type then
     begin
     MFN_NUMBER:= 0;
     CLEARBOX(8,1,17,75,0);
     MESSAGE(10,7,'Sorry, You gave a wrong record type !',
             18,7,'Type any key to return to the menu',18,41,'','halt');
     end;
end;

if seq='X' then scode:= seq;
if seq='X' then MFN_NUMBER:= 0;
end; { function }


{ ------- Body of program -------- }
begin
CLEAR;
MESSAGE(3,7,'*** COPYING A PROJECT RECORD ***',1,1,'',1,1,'','');
```

```
wks_name:= 'proj.fmt';
r_type:='P';
m_l:= MFN_NUMBER(r_type,sc);

      if m_l>0 and sc<>'X' then
         begin
           d:= RECORD(m_l);
           TRANSFER(m_l,wks_name);
           n_r:= MAXMFN; n_r:= n_r-1;    { new record number }
           d:= WORKSHEET('proj');
           AUTOTYPE('M^M^M');
           d:= MODIFY(n_r);
         end; {m_l>0 and sc<>'X'}

q:= ' ';
end.
```

```
Program DISP(d: string) [menu];
{-------------------------------------------------------------}
{     This program selects one of the display formats         }
{     A moving arrow and highlighting are used as markers     }
{     The program is invoked from the menu EXGEN, option D    }
{-------------------------------------------------------------}

                    { **** by M.Muraszkiewicz, Oct. 10, 1991 ***** }

var     flag, fn, ind, nx_r_tg, dummy              :real;
        r_num, frc, lrc, n_hit, i, l_count         :real;
        lin, col, hi, wi, le, poz                  :real;
        code, fornm, line, s, dum                  :string;

Procedure MESSAGE(lin1,col1:real;text1:string;
                  lin2,col2:real;text2:string;
                  lin3,col3:real;text3:string;
                  halt:string);

{-----------------------------------------------------}
{         Displays the messages                       }
{-----------------------------------------------------}

var du   :string;

begin
cursor(lin1,col1);
writeln(text1);
cursor(lin2,col2);
writeln(text2);
cursor(lin3,col3);
write(text3);
if halt='halt' then du:= INKEY;

end; {procedure}


Function SELECT(stlin,stcol,high,wide,len :real;
                str,scode:string): real;

{-------------------------------------------------------------}
{   This function allows for selecting an option by means     }
{   of an arrow and highlighting.                             }
{      Input paramaters:  stlin, stcol, high, wide, len       }
{      Output parameters: SELECT, str                         }
{-------------------------------------------------------------}

var  sc, i, k :real;
     d        :string;

begin
i:= 0; k:= 0; str:= '';
CHATTR(2,stlin+1,stcol+1,len);                { put new highlight  }
repeat
```

```
CURSOR(stlin+1+i,stcol+wide);
{writeln('<----'); }                                    { draw  new arrow    }
ATTR(' ',0,26,80,1);                                   { hide a cursor      }
sc:= KBDKEY(str);
if sc = 80 then i:= i+1;                               { down arrow was met }
if sc = 72 then i:= i-1;                               { up arrow was met   }
if i > high-3 then i:= 0;                              { skip to the top    }
if i < 0 then i:=high-3;                               { skip to the bottom }
if sc=72 or sc=80 then
   begin
   CLEARBOX(stlin+1+k,stcol+wide,1,5,0);               { remove old arrow   }
   CHATTR(0,stlin+1+k,stcol+1,len);                    { remove old hlight  }
   CHATTR(2,stlin+1+i,stcol+1,len);                    { put new highlight  }
   end;
k:= i;
until sc=28  or              { ENTER }
      sc=73  or              { PgUp  }
      sc=81  or              { PgDn  }
      str='x' or str='X';    { X was met }
scode:= ENCINT(sc,2);
SELECT:= k;
end;


Function ITEM(stlin,stcol,high,wide,len :real;
              text1,
              text2,
              text3:string):real;

{-------------------------------------------------------}
{     This program selects one of the items             }
{     ITEM returns row number of the selected item      }
{     0 is returned if x or X met                       }
{-------------------------------------------------------}

var    sop, lin, col, hi, wi, le          :real;
       scode, str                         :string;

begin

{ -------- draw a box -----------}
CLEARBOX(stlin,stcol,high,wide,0);
BOX(stlin,stcol,high,wide,2);
ATTR(' ',2,stlin+1,stcol+1,len);

{ --------- write the items in the box ------- }
MESSAGE(stlin+1,stcol+1,text1,1,1,'',1,1,'','');
MESSAGE(stlin+2,stcol+1,text2,1,1,'',1,1,'','');
MESSAGE(stlin+3,stcol+1,text3,1,1,'',1,1,'','');
MESSAGE(stlin+high-1,stcol+wide-7,'X-Exit',1,1,'',1,1,'','');
MESSAGE(22,1,'Select your item by means of  , ',
        23,1,'Type ENTER to confirm you choice',
        24,1,'Strike X to quit','');
```

```
sop:= SELECT(stlin,stcol,high,wide,len,str,scode);   { pick up item }
UC(str);

ITEM:= sop+1;
if str = 'X' then ITEM:=0;
end; {function}


     { ---------------------- Body of Program ---------------------- }
begin
CLEAR;
d:= '.D';

{------- draw boxes an list items and select one ----}
lin:= 12; col:= 15; hi:= 5; wi:= 20; le:= wi-2;
poz:= ITEM(lin,col,hi,wi,le,
           '  Organisation',
           '  Project',
           '  Both');
if poz=0 then d:= ' ';
if poz<> 0 then
begin
   CASE poz OF
   1: begin fornm:='ENTRE'; GETFMT('@':fornm); CLEAR; end;
   2: begin fornm:='PROJ';  GETFMT('@':fornm); CLEAR; end;
   3: begin
      CLEAR;
      l_count:= 1;
      nx_r_tg:= 0;
      flag:=0;
      i:= 1;
      n_hit:= SETPOS(0,0);                        { number of hits }
      if (n_hit>0) then
         begin
         WHILE i<=n_hit DO
              begin
              r_num:= SETPOS(0,i);                  { record number }
              ind:= RECORD(r_num);                  { get the record }
              i:= i+1;
              if (ind<=0) then          { was the record deleted ? }
                  begin
     { ---- get type of record --- }                              { no }
                  fn:= FIELDN(3,1);
                  code:= FIELD(fn);           { get the field value }

     { --------- formats --------- }
                  if code='I' then fornm:='ENTRE';
                  if code='P' then fornm:='PROJ';

                  GETFMT('@':fornm);
     { --------- display --------- }
                  frc:= FORMAT(79);
         if frc=0 then
            begin
```

```
                lrc:= NXTLINE(line);
                while lrc=0 do
                  begin
                    writeln(line);
                    l_count:= l_count+1;
                    lrc:= NXTLINE(line);
                    if lrc<>0 then flag:=1;
                    if l_count>22 then
                       begin
                       l_count:=1; CURSOR(24,1); write('More...');
                       le:= KBDKEY(dum);          { looking for ESC }
                       CLEAR; CURSOR(1,1);
                       if le=1 then
                          begin
                          lrc:=1;
                          i:=n_hit+2;           { ESC met }
                          end;
                       if nx_r_tg=1 then
                          begin
                          lrc:=1;
                          i:= i-1;
                          flag:=0;
                          end;
                       end;
                  end; {while}
              end {frc=0}
                  else writeln('   Format error ',frc:1);
            nx_r_tg:=flag;
    end; {WHILE}
    end {d<=0};
  end;  {n_hit>0}
write('*** End of display ***'); s:= INKEY;
d:= ' ';                          { return to menu }
end; {3 case}
end; {case}
end; {poz}
end.
```

```
Program ENTFOR(b: string) [menu];
{     This program selects one of the ENTRE display formats          }
{     A moving arrow and highlighting are used as markers            }
{     The program is invoked from the menu EXGEN, option B           }

                    { **** by M.Muraszkiewicz, Sept. 14, 1991 **** }

var     sop, lin, col, hi, wi, le              :real;
        str, fornm                             :string;

Function SELECT(stlin,stcol,high,wide,len :real;str :string): real;
    {  This function allows for selecting an option by means    }
    {  of an arrow and highlighting.                            }
    {    Input paramaters:  stlin, stcol, high, wide, len       }
    {    Output parameters: SELECT, str                         }
var sc, i, k :real;
begin
i:= 0; k:= 0; str:= '';
repeat
CURSOR(stlin+1+i,stcol+wide);
writeln('<----');                                  { draw  new arrow     }
ATTR(' ',0,26,80,1);                               { hide a cursor       }
sc:= KBDKEY(str);
if sc = 80 then i:= i+1;                           { down arrow was met  }
if sc = 72 then i:= i-1;                           { up arrow was met    }
if i > high-3 then i:= 0;                          { skip to the top     }
if i < 0 then i:=high-3;                           { skip to the bottom  }
if sc=72 or sc=80 then
   begin
   CLEARBOX(stlin+1+k,stcol+wide,1,5,0);           { remove old arrow    }
   CHATTR(0,stlin+1+k,stcol+1,len);                { remove old hlight   }
   CHATTR(2,stlin+1+i,stcol+1,len);                { put new highlight   }
   end;
k:= i;
until sc=28 or str='x' or str='X';                 { ENTER or X was met  }
SELECT:= k;
end;

Procedure INDEX_BOX(stlin,stcol,high,wide,len :real);
{  Draws a box and fills it out with indexes names  }
begin
BOX(stlin,stcol,high,wide,2);
ATTR(' ',2,stlin+1,stcol+1,len);                   { 2 - hlight  }
CURSOR(stlin+1,stcol+1); write('  Organisation');
CURSOR(stlin+2,stcol+1); write('  Project');
CURSOR(stlin+high+1,stcol+4); write('X - Exit');
end;

Procedure MESSAGE;
begin
CURSOR(8,18); write('Display Formats');
CURSOR(22,1);
write('Select your format by means of arrows ');
CURSOR(23,1); write('Type ENTER to confir you choice');
```

```
CURSOR(24,1); write('Strike X to quit');
end;
                { -------- Body of Program -------- }
begin
b:= ' ';
CLEAR;                                          { clear screen }
MESSAGE;                                        { display msgs }
CURSOR(4,3);

   { ---------- Display and pick up a format name ---------- }
   lin:= 9; col:= 15; hi:= 4; wi:= 20; le:= wi-2;
   INDEX_BOX(lin,col,hi,wi,le); CURSOR(lin-1,col+1); {draw a box + opt.}
   sop:= SELECT(lin,col,hi,wi,le,str);              { pick up by <--- }
   if str <> 'x' and str <> 'X' then
      begin                                         { X - Exit not met }
      { --------- formats --------- }
      if sop = 0 then fornm:='ENTRE';
      if sop = 1 then fornm:='PROJ';
      GETFMT('@':fornm);
      b:= '.B';
      end;
CLEARBOX(22,1,3,80,0);
end.
```

```
Program ENTEDI(l: string) [menu];
{     This program selects a worksheet for the ENTRE database       }
{     A moving arrow and highlighting are used as markers           }
{     The program is invoked from the menu EXE1, option E           }

                    { **** by M.Muraszkiewicz, Oct. 14, 1991 **** }

var    m_l, d, lin, col, hi, wi, le          :real;
       sheet, type                           :string;


Procedure MESSAGE(lin1,col1:real;text1:string;
                  lin2,col2:real;text2:string;
                  lin3,col3:real;text3:string;
                  halt:string);

{-------------------------------------------------------}
{         Displays the messages                          }
{-------------------------------------------------------}

var du   :string;

begin
cursor(lin1,col1);
writeln(text1);
cursor(lin2,col2);
writeln(text2);
cursor(lin3,col3);
write(text3);
if halt='halt' then du:= INKEY;

end; {procedure}

Function MFN_NUMBER:real;

{-----------------------------------------------------------}
{   Asks for the number of the record to be transferred     }
{   returns 0 if X or x met or wrong record type            }
{-----------------------------------------------------------}

var    m_l, max, d    :real;
       seq, typ       :string;
begin
m_l:=0;
CLEARBOX(22,1,2,80,0);
CURSOR(24,1); writeln('Type X [Enter] to leave without moving the record');
repeat
  CLEARBOX(7,1,1,80,0); CURSOR(7,7);
  write('Pls, specify MFN of the record to be edited: ');
  readln(seq); m_l:=val(seq);
  UC(seq);                                  { upper case }
  max:= MAXMFN;
  CLEARBOX(9,1,1,80,0); CURSOR(9,7);
  if (m_l>=max) then
```

```
     begin
     CLEARBOX(9,1,1,80,0); CURSOR(9,7);
     writeln('Sorry, your MFN is wrong !   Try again, pls');
     end;
until ( (m_1<max) AND (m_1>0) OR (seq='X'));
MFN_NUMBER:= m_1;

if seq='X' then MFN_NUMBER:= 0;
end; { function }


     { --- Body of Program --- }
begin
CLEAR;

m_1:= MFN_NUMBER;
if m_1 > 0 then
begin
  d:= RECORD(m_1);
  type:= FIELD(FIELDN(3,1));   { get the record type }
  if type='I' or type='P' then
     begin
     CASE type OF
     'I': sheet:='ENTRE';
     'P': sheet:='PROJ';
     end;
     d:= WORKSHEET(sheet);
     d:= MODIFY(m_1);
     end
      else MESSAGE(12,4,'Record corrupted',
                   14,4,'Press any key to continue',14,39,'','halt');
end; {m_1>0}
l:= ' ';
end.
```

```
Program ENTFOR(f: string) [menu];
{     This program selects one of the ENTRE display formats          }
{     A moving arrow and highlighting are used as markers            }
{     The program is invoked from the menu EFGEN, option F           }

                    { **** by M.Muraszkiewicz, Sept. 14, 1991 **** }

var    sop, lin, col, hi, wi, le          :real;
       str, fornm                         :string;

Function SELECT(stlin,stcol,high,wide,len :real;str :string): real;
    {  This function allows for selecting an option by means    }
    {  of an arrow and highlighting.                            }
    {    Input paramaters:  stlin, stcol, high, wide, len       }
    {    Output parameters: SELECT, str                         }
var sc, i, k :real;
begin
i:= 0; k:= 0; str:= '';
repeat
CURSOR(stlin+1+i,stcol+wide);
writeln('<----');                              { draw  new arrow    }
ATTR(' ',0,26,80,1);                           { hide a cursor      }
sc:= KBDKEY(str);
if sc = 80 then i:= i+1;                       { down arrow was met }
if sc = 72 then i:= i-1;                       { up arrow was met   }
if i > high-3 then i:= 0;                      { skip to the top    }
if i < 0 then i:=high-3;                       { skip to the bottom }
if sc=72 or sc=80 then
   begin
   CLEARBOX(stlin+1+k,stcol+wide,1,5,0);       { remove old arrow   }
   CHATTR(0,stlin+1+k,stcol+1,len);            { remove old hlight  }
   CHATTR(2,stlin+1+i,stcol+1,len);            { put new highlight  }
   end;
k:= i;
until sc=28 or str='x' or str='X';             { ENTER or X was met }
SELECT:= k;
end;

Procedure INDEX_BOX(stlin,stcol,high,wide,len :real);
{  Draws a box and fills it out with indexes names  }
begin
BOX(stlin,stcol,high,wide,2);
ATTR(' ',2,stlin+1,stcol+1,len);                   { 2 - hlight  }
CURSOR(stlin+1,stcol+1); write('  Organisation');
CURSOR(stlin+2,stcol+1); write('  Project');
CURSOR(stlin+high+1,stcol+4); write('X - Exit');
end;

Procedure MESSAGE;
begin
CURSOR(8,18); write('Display Formats');
CURSOR(22,1);
write('Select your format by means of arrows  ');
CURSOR(23,1); write('Type ENTER to confir you choice');
```

```
CURSOR(23,1); write('Strike X to quit');
end;
                { -------- Body of Program -------- }
begin
CLEAR;                                          { clear screen }
MESSAGE;                                        { display msgs }
CURSOR(4,3);

   { ---------- Display and pick up a format name ---------- }
   lin:= 9; col:= 15; hi:= 4; wi:= 20; le:= wi-2;
   INDEX_BOX(lin,col,hi,wi,le); CURSOR(lin-1,col+1); {draw a box + opt.}
   sop:= SELECT(lin,col,hi,wi,le,str);              { pick up by <--- }
   if str <> 'x' and str <> 'X' then
      begin                                         { X - Exit not met }
      { --------- formats --------- }
      if sop = 0 then fornm:='ENTRE';
      if sop = 1 then fornm:='PROJ';
      GETFMT('@':fornm);
      end;
   f:= ' ';                                          { return to menu }
end.
```

```
Program ENTIND(s: string) [menu];
{     This program selects one of the ENTRE Indexes for printing    }
{     A moving arrow and highlighting are used as markers           }
{     The program is invoked from the menu FXPRT, option S          }

                        { **** by M.Muraszkiewicz, Oct. 2, 1991 **** }

var     sop1, sop2, lin, col, hi, wi, le      :real;
        str, sheet                            :string;


Procedure DSPL(nrow,ncol: real; text: string);
begin
CURSOR(nrow,ncol); WRITE(text);
end;

Function SELECT(stlin,stcol,high,wide,len :real;str :string): real;
    {  This function allows for selecting an option by means   }
    {  of an arrow and highlighting.                           }
    {    Input paramaters:  stlin, stcol, high, wide, len      }
    {    Output parameters: SELECT, str                        }
var sc, i, k :real;
begin
i:= 0; k:= 0; str:= '';
repeat
CURSOR(stlin+1+i,stcol+wide);
writeln('<----');                                  { draw  new arrow    }
ATTR(' ',0,26,80,1);                               { hide a cursor      }
sc:= KBDKEY(str);
if sc = 80 then i:= i+1;                           { down arrow was met }
if sc = 72 then i:= i-1;                           { up arrow was met   }
if i > high-3 then i:= 0;                          { skip to the top    }
if i < 0 then i:=high-3;                           { skip to the bottom }
if sc=72 or sc=80 then
   begin
   CLEARBOX(stlin+1+k,stcol+wide,1,5,0);           { remove old arrow   }
   CHATTR(0,stlin+1+k,stcol+1,len);                { remove old hlight  }
   CHATTR(2,stlin+1+i,stcol+1,len);                { put new highlight  }
   end;
k:= i;
until (sc=28) or (str='x') or (str='X');           { ENTER or X was met }
SELECT:= k;
end;

Procedure INDEX_BOX1(stlin,stcol,high,wide,len :real);
{  Draws a box and fills it out with 2 options  }
begin
BOX(stlin,stcol,high,wide,1);
ATTR(' ',2,stlin+1,stcol+1,len);                   { 2 - hlight  }
DSPL(stlin+1,stcol+1,'Hits');
DSPL(stlin+2,stcol+1,'Whole base');
DSPL(stlin+high+1,stcol+4,'X - Exit');
end;
```

```
Procedure INDEX_BOX2(stlin,stcol,high,wide,len :real);
{  Draws a box and fills it out with indexes names  }
begin
BOX(stlin,stcol,high,wide,1);
ATTR(' ',2,stlin+1,stcol+1,len);                    { 2 - hlight  }
DSPL(stlin+1,stcol+1,'List of References');
DSPL(stlin+2,stcol+1,'Subjet Index');
DSPL(stlin+high+1,stcol+4,'X - Exit');
end;

Procedure MESSAGE;
begin
DSPL(2,25,'ENTRE Index');
CURSOR(22,1);
write('Select your format by means of arrows  ');
CURSOR(23,1); write('Type ENTER to confir you choice');
CURSOR(23.1); write('Strike X to quit');

end;
                { -------- Body of Program -------- }
begin
CLEAR;                                              { clear screen }
MESSAGE;                                            { display msgs }
CURSOR(5,5);

       { --------- Hits or Whole database ? ---------- }
write('Do you want to create index from hits or whole database?');
lin:= 9; col:= 29; hi:= 4; wi:= 17; le:= wi-2;
s:= ' ';                                            { return to print menu }
INDEX_BOX1(lin,col,hi,wi,le);                       { draw a box + options }
sop1:= SELECT(lin,col,hi,wi,le,str);                { pick up by <----     }
CLEAR;
if (str <> 'x') and (str <> 'X') then
   begin                                                 { X - Exit not met }
   { ---------- Display and pick up ENTRE Indexes ---------- }
   MESSAGE; lin:= 8; col:= 23; hi:= 4; wi:= 35; le:= wi-2;
   INDEX_BOX2(lin,col,hi,wi,le); CURSOR(lin-1,col+1); {draw a box + opt.}
   if sop1 = 0 then write('HITS');
   if sop1 = 1 then write('TOUTE LA BASE');
   sop2:= SELECT(lin,col,hi,wi,le,str);                 { pick up by <--- }
   if (str <> 'x') and (str <> 'X') then
      begin                                            { X - Exit not met }
      { ------------------ hits ----------------- }
      if sop1 = 0 and sop2 = 0 then sheet:='fye1h';
      if sop1 = 0 and sop2 = 1 then sheet:='fye2h';
      { ------------- whole database ------------ }
      if sop1 = 1 and sop2 = 0 then sheet:='fye1b';
      if sop1 = 1 and sop2 = 1 then sheet:='fye2b';
      AUTOTYPE(sheet);
      CLEARMSG; CURSOR(22,21); write(sheet);
      s:= '.S';                                  { return & call a worksheet }
      end;
   end;
end.
```

```
Program ENTWSH(l: string) {menu};
{     This program selects a worksheet and creates a record          }
{     A moving arrow and highlighting are used as markers            }
{     The program is invoked from the menu EXE1, option N            }

                    { **** by M.Muraszkiewicz, Sept. 14, 1991 **** }

var    poz, d, lin, col, hi, wi, le          :real;
       sheet                                 :string;


Procedure MESSAGE(lin1,col1:real;text1:string;
                  lin2,col2:real;text2:string;
                  lin3,col3:real;text3:string;
                  halt:string);

{-------------------------------------------------------}
{         Displays the messages                         }
{-------------------------------------------------------}

var du   :string;

begin
cursor(lin1,col1);
writeln(text1);
cursor(lin2,col2);
writeln(text2);
cursor(lin3,col3);
write(text3);
if halt='halt' then du:= INKEY;

end; {procedure}


Function SELECT(stlin,stcol,high,wide,len :real;
                str,scode:string): real;

{-----------------------------------------------------------}
{   This function allows for selecting an option by means   }
{   of an arrow and highlighting.                           }
{      Input paramaters:  stlin, stcol, high, wide, len     }
{      Output parameters: SELECT, str                       }
{-----------------------------------------------------------}

var  sc, i, k :real;
     d        :string;

begin
i:= 0; k:= 0; str:= '';
CHATTR(2,stlin+1,stcol+1,len);              { put new highlight  }
repeat
CURSOR(stlin+1+i,stcol+wide);
{writeln('<----'); }                              { draw  new arrow    }
ATTR(' ',0,26,80,1);                            { hide a cursor      }
```

```
sc:= KBDKEY(str);
if sc = 80 then i:= i+1;                          { down arrow was met }
if sc = 72 then i:= i-1;                          { up arrow was met   }
if i > high-3 then i:= 0;                         { skip to the top    }
if i < 0 then i:=high-3;                          { skip to the bottom }
if sc=72 or sc=80 then
   begin
   CLEARBOX(stlin+1+k,stcol+wide,1,5,0);          { remove old arrow   }
   CHATTR(0,stlin+1+k,stcol+1,len);               { remove old hlight  }
   CHATTR(2,stlin+1+i,stcol+1,len);               { put new highlight  }
   end;
k:= i;
until sc=28  or            { ENTER }
      sc=73  or            { PgUp  }
      sc=81  or            { PgDn  }
      str='x' or str='X';  { X was met }
scode:= ENCINT(sc,2);
SELECT:= k;
end;


Function ITEM(stlin,stcol,high,wide,len :real;
              text1,
              text2,
              text3:string):real;

{-------------------------------------------------------}
{     This program selects one of the items             }
{     ITEM returns row number of the selected item      }
{     0 is returned if x or X met                       }
{-------------------------------------------------------}

var    sop, lin, col, hi, wi, le          :real;
       scode, str                         :string;

begin

{ -------- draw a box -----------}
CLEARBOX(stlin,stcol,high,wide,0);
BOX(stlin,stcol,high,wide,2);
ATTR(' ',2,stlin+1,stcol+1,len);

{ --------- write the items in the box ------- }
MESSAGE(stlin+1,stcol+1,text1,1,1.'',1,1,'','');
MESSAGE(stlin+2,stcol+1,text2,1,1,'',1,1,'','');
MESSAGE(stlin+3,stcol+1,text3,1,1,'',1,1,'','');
MESSAGE(stlin+high-1,stcol+wide-7,'X-Exit',1,1,'',1,1,'','');
MESSAGE(22,1,'Select your item by means of  , ',
        23,1,'Type ENTER to confirm you choice',
        24,1,'Strike X to quit','');

sop:= SELECT(stlin,stcol,high,wide,len,str,scode);   { pick up item }
UC(str);
```

```
ITEM:= sop+1;
if str = 'X' then ITEM:=0;
end; {function}


     { ---------------------- Body of Program -------------------- }
begin
CLEAR;

{---·--- draw boxes an list items and select one ----}
lin:= 12; col:= 15; hi:= 4; wi:= 20; le:= wi-2;
poz:= ITEM(lin,col,hi,wi,le,
           '  Organisation',
           '  Project','');
{ --------- worksheets --------- }
if poz <> 0 then
   begin
     CASE poz OF
     1: sheet:='ENTRE';
     2: sheet:='PROJ';
     end;
     d:= WORKSHEET(sheet);
     l:= '.N';
   end;
if poz=0 then l:= ' ';
end.
```

```
Program LOGIND(s: string) [menu];
{     This program selects one of the LOGI  Indexes for printing    }
{     A moving arrow and highlighting are used as markers           }
{     The program is invoked from the menu FXPRT. option S          }

                    { **** by M.Muraszkiewicz, Oct. 2, 1991 **** }

var    sop1, sop2, lin, col, hi, wi, le     :real;
       str, sheet                           :string;


Procedure DSPL(nrow,ncol: real; text: string);
begin
CURSOR(nrow,ncol); WRITE(text);
end;

Function SELECT(stlin,stcol,high,wide,len :real;str :string): real;
    {  This function allows for selecting an option by means   }
    {  of an arrow and highlighting.                           }
    {    Input paramaters:  stlin, stcol, high, wide, len      }
    {    Output parameters: SELECT, str                        }
var sc, i, k :real;
begin
i:= 0; k:= 0; str:= '';
repeat
CURSOR(stlin+1+i,stcol+wide);
writeln('<----');                                  { draw  new arrow    }
ATTR(' ',0,26,80,1);                               { hide a cursor      }
sc:= KBDKEY(str);
if sc = 80 then i:= i+1;                           { down arrow was met }
if sc = 72 then i:= i-1;                           { up arrow was met   }
if i > high-3 then i:= 0;                          { skip to the top    }
if i < 0 then i:=high-3;                           { skip to the bottom }
if sc=72 or sc=80 then
   begin
   CLEARBOX(stlin+1+k,stcol+wide,1,5,0);           { remove old arrow   }
   CHATTR(0,stlin+1+k,stcol+1,len);                { remove old hlight  }
   CHATTR(2,stlin+1+i,stcol+1,len);                { put new highlight  }
   end;
k:= i;
until (sc=28) or (str='x') or (str='X');           { ENTER or X was met }
SELECT:= k;
end;

Procedure INDEX_BOX1(stlin,stcol,high,wide,len :real);
{  Draws a box and fills it out with 2 options  }
begin
BOX(stlin,stcol,high,wide,1);
ATTR(' ',2,stlin+1,stcol+1,len);                   { 2 - hlight  }
DSPL(stlin+1,stcol+1,'Hits');
DSPL(stlin+2,stcol+1,'Whole base');
DSPL(stlin+high+1,stcol+4,'X - Exit');
end;
```

```
Procedure MESSAGE;
begin
DSPL(2,24,'List of references of the LOGI database');
CURSOR(22,1);
write('Select your option by arrows ');
DSPL(23,1,'Type ENTER to confirm your choice');
DSPL(24,1,'Press X to quit');
end;
                { -------- Body of Program -------- }
begin
CLEAR;                                          { clear screen }
MESSAGE;                                        { display msgs }
CURSOR(5,5);

       { --------- Hits or Whole database ? ----------- }
write('Do you want hits or the whole database?');
lin:= 9; col:= 29; hi:= 4; wi:= 17; le:= wi-2;
s:= ' ';                                        { return to print menu }
INDEX_BOX1(lin,col,hi,wi,le);                   { draw a box + options }
sop1:= SELECT(lin,col,hi,wi,le,str);            { pick up by <----     }

CLEAR;
if (str <> 'x') and (str <> 'X') then
   begin                                           { X - Exit not met }
      { ------------------ hits ----------------- }
      if sop1 = 0 then sheet:='fyl1h';
      if sop1 = 1 then sheet:='fyl1b';
      AUTOTYPE(sheet);
      CLEARMSG; CURSOR(22,21); write(sheet);
      s:= '.S';                                 { return & call a worksheet }
   end;
end.
```

```
Program LOGIND(s: string) [menu];
{     This program selects one of the LOGI  Indexes for printing    }
{     A moving arrow and highlighting are used as markers           }
{     The program is invoked from the menu FXPRT, option S          }

                    { **** by M.Muraszkiewicz, Oct. 2, 1991 **** }

var    sop1, sop2, lin, col, hi, wi, le     :real;
       str, sheet                           :string;


Procedure DSPL(nrow,ncol: real; text: string);
begin
CURSOR(nrow,ncol); WRITE(text);
end;

Function SELECT(stlin,stcol,high,wide,len :real;str :string): real;
    {  This function allows for selecting an option by means   }
    {  of an arrow and highlighting.                           }
    {    Input paramaters:  stlin, stcol, high, wide, len      }
    {    Output parameters: SELECT, str                        }
var sc, i, k :real;
begin
i:= 0; k:= 0; str:= '';
repeat
CURSOR(stlin+1+i,stcol+wide);
writeln('<----');                              { draw  new arrow    }
ATTR(' ',0,26,80,1);                           { hide a cursor      }
sc:= KBDKEY(str);
if sc = 80 then i:= i+1;                       { down arrow was met }
if sc = 72 then i:= i-1;                       { up arrow was met   }
if i > high-3 then i:= 0;                      { skip to the top    }
if i < 0 then i:=high-3;                       { skip to the bottom }
if sc=72 or sc=80 then
   begin
   CLEARBOX(stlin+1+k,stcol+wide,1,5,0);       { remove old arrow   }
   CHATTR(0,stlin+1+k,stcol+1,len);            { remove old hlight  }
   CHATTR(2,stlin+1+i,stcol+1,len);            { put new highlight  }
   end;
k:= i;
until (sc=28) or (str='x') or (str='X');       { ENTER or X was met }
SELECT:= k;
end;

Procedure INDEX_BOX1(stlin,stcol,high,wide,len :real);
{  Draws a box and fills it out with 2 options  }
begin
BOX(stlin,stcol,high,wide,1);
ATTR(' ',2,stlin+1,stcol+1,len);                   { 2 - hlight  }
DSPL(stlin+1,stcol+1,'Hits');
DSPL(stlin+2,stcol+1,'Whole database');
DSPL(stlin+high+1,stcol+4,'X - Exit');
end;
```

```
Procedure MESSAGE;
begin
DSPL(2,24,'List of references of the LOGI database');
CURSOR(22,1);
write('Select your option by arrows');
DSPL(23,1,'Type ENTER to confirm your choice');
DSPL(24,1,'Press X to quit');
end;

                { -------- Body of Program -------- }
begin
CLEAR;                                          { clear screen }
MESSAGE;                                        { display msgs }
CURSOR(5,5);

       { --------- Hits or Whole database ? ----------- }
write('Do you want hits or the whole database?');
lin:= 9; col:= 29; hi:= 4; wi:= 17; le:= wi-2;
s:= ' ';                                        { return to print menu }
INDEX_BOX1(lin,col,hi,wi,le);                   { draw a box + options }
sop1:= SELECT(lin,col,hi,wi,le,str);            { pick up by <----     }

CLEAR;
if (str <> 'x') and (str <> 'X') then
   begin                                                { X - Exit not met }
      { ------------------ hits ----------------- }
      if sop1 = 0 then sheet:='fyl1h';
      if sop1 = 1 then sheet:='fyl1b';
      AUTOTYPE(sheet);
      CLEARMSG; CURSOR(22,21); write(sheet);
      s:= '.S';                                 { return & call a worksheet }
   end;
end.
```

```
Program PFTCRE;
{-------------------------------------------------------------}
{     This program generates draft xxxxx.pft table            }
{     from the xxxxx.fdt table                                }
{     dbn.par specifies the location of xxxxx.fdt and xxxxx.pft }
{                                                             }
{     NOTE ! Subfields are not supported                      }
{            xxxxx.pft file is assumed to exist               }
{            X or x cannot be the name of pft file
{-------------------------------------------------------------}

                  { **** by M.Muraszkiewicz, Aug. 18, 1991 **** }


var   pos1, pos2, pos3, pos4                    :real;
      j, i, pft_ex, fdt_ex, siz, len            :real;
      line, f_name, f_val, f_ln, f_pat, f_rep   :string;
      sufx, pft_line, znak, dum                 :string;
      fdt_name, pft_name                        :string;
      col, hyp                                  :string;

Function TABLE_NAME(l:real;type:string):string;

{-----------------------------------------------------------}
{            Asks for the name of file                      }
{            Returns 0 if X or x met                        }
{-----------------------------------------------------------}

var   seq   :string;

begin
CURSOR(3,7);
write('*** This programs creates draft PFT table ****');
CURSOR(24,1);
writeln('Type X [Enter] to leave without moving the record');
CURSOR(1,7);
write('Pls, specify the name (prefix only) of the ',type,' table: ');
readln(seq); UC(seq);
TABLE_NAME:= seq+'.'+type;
if seq='X' then TABLE_NAME:= '0';
end; { function }
                { ----- Body of program ----- }
begin

{------------------PARAMETERS ----------------}
col:= '30';
hyp:= '29';
{--------------------------------------------}

CLEAR;
fdt_name:= TABLE_NAME(7,'FDT');
if fdt_name<>'0'then
begin
pft_name:= TABLE_NAME(9,'PFT');
```

```
if pft_name<>'O' then
begin

fdt_ex:= FILEXIST(path('dbn',10)+fdt_name);
pft_ex:= FILEXIST(path('dbn',10)+pft_name);

if ((fdt_ex=0) and (pft_ex=0)) then
   begin
   ASSIGN('INP',path('dbn',10)+fdt_name);
   ASSIGN('OUT',path('dbn',10)+pft_name);
   FOR i:= 1 TO 4 DO  readln(INP,line);

     clear;

   WHILE NOT(EOF(INP)) DO
   begin
     readln(INP,line);          { get a line from fdt }
     siz:= SIZE(line);
     f_name:= SUBSTR(line,1,30);
     i:= 30;
     REPEAT
      znak:= SUBSTR(f_name,i,1);
      i:= i-1;
     UNTIL (znak<>' 'or i=0);
     i:= i+1;
     f_name:= SUBSTR(f_name,1,i);               { field name   }

     pos1:= POSITION(line,' ',51);
     f_val:= SUBSTR(line,51,pos1-51);           { field value  }

     pos2:= POSITION(line,' ',pos1+1);
     f_ln:= SUBSTR(line,pos1+1,pos2-pos1);      { field length }

     pos3:= POSITION(line,' ',pos2+1);
     f_pat:= SUBSTR(line,pos2+1,pos3-pos2);      { field type  }

     f_rep:= SUBSTR(line,pos3+1,siz-pos3);       { repetab. tag }

  {--------- make pft format line ------------}
     sufx:= 'v'+f_val+'('+hyp+','+hyp+')/';
     if f_rep='1' then sufx:= '(v'+f_val+'('+hyp+','+hyp+')+; ;)/';
     pft_line:= '"'+f_name+':"'+'C'+col+'D'+f_val+','+sufx;

     len:= SIZE(pft_line);
     FOR j:=1 TO 80-len DO
     pft_line:= pft_line+',';          { adjust lines with commas }

     writeln(OUT,pft_line:80);

   end; {while}
 end;
end;
end;
end.
```

```
Program PREIND(s: string) [menu];
{     This program selects one of the PRETS Indexes for printing    }
{     A moving arrow and highlighting are used as markers           }
{     The program is invoked from the menu FXPRT, option S          }

                    { **** by M.Muraszkiewicz, Oct. 2, 1991 **** }

var    sop1, sop2, lin, col, hi, wi, le     :real;
       str, sheet                           :string;


Procedure DSPL(nrow,ncol: real; text: string);
begin
CURSOR(nrow,ncol); WRITE(text);
end;

Function SELECT(stlin,stcol,high,wide,len :real;str :string): real;
    {  This function allows for selecting an option by means   }
    {  of an arrow and highlighting.                           }
    {    Input paramaters:  stlin, stcol, high, wide, len      }
    {    Output parameters: SELECT, str                        }
var sc, i, k :real;
begin
i:= 0; k:= 0; str:= '';
repeat
CURSOR(stlin+1+i,stcol+wide);
writeln('<----');                                  { draw  new arrow    }
ATTR(' ',0,26,80,1);                               { hide a cursor      }
sc:= KBDKEY(str);
if sc = 80 then i:= i+1;                           { down arrow was met }
if sc = 72 then i:= i-1;                           { up arrow was met   }
if i > high-3 then i:= 0;                          { skip to the top    }
if i < 0 then i:=high-3;                           { skip to the bottom }
if sc=72 or sc=80 then
   begin
   CLEARBOX(stlin+1+k,stcol+wide,1,5,0);           { remove old arrow   }
   CHATTR(0,stlin+1+k,stcol+1,len);                { remove old hlight  }
   CHATTR(2,stlin+1+i,stcol+1,len);                { put new highlight  }
   end;
k:= i;
until (sc=28) or (str='x') or (str='X');           { ENTER or X was met }
SELECT:= k;
end;

Procedure INDEX_BOX1(stlin,stcol,high,wide,len :real);
{  Draws a box and fills it out with 2 options  }
begin
BOX(stlin,stcol,high,wide,1);
ATTR(' ',2,stlin+1,stcol+1,len);                   { 2 - hlight  }
DSPL(stlin+1,stcol+1,'Hits');
DSPL(stlin+2,stcol+1,'Whole base');
DSPL(stlin+high+1,stcol+4,'X - Exit');
end;
```

```
Procedure MESSAGE;
begin
DSPL(2,24,'List of references PRETS');
CURSOR(22,1);
write('Select your format by means of arrows  ');
CURSOR(23,1); write('Type ENTER to confir you choice');
CURSOR(23,1); write('Strike X to quit');

end;
                { -------- Body of Program -------- }
begin
CLEAR;                                          { clear screen }
MESSAGE;                                        { display msgs }
CURSOR(5,5);

       { --------- Hits or Whole database ? ----------- }
write('Do you want to create index from hits or the whole database?');
lin:= 9; col:= 29; hi:= 4; wi:= 17; le:= wi-2;
s:= ' ';                                        { return to print menu }
INDEX_BOX1(lin,col,hi,wi,le);                   { draw a box + options }
sop1:= SELECT(lin,col,hi,wi,le,str);            { pick up by <----     }

CLEAR;
if (str <> 'x') and (str <> 'X') then
   begin                                           { X - Exit not met }
      { ------------------ hits ----------------- }
      if sop1 = 0 then sheet:='fyp1h';
      if sop1 = 1 then sheet:='fyp1b';
      AUTOTYPE(sheet);
      CLEARMSG; CURSOR(22,21); write(sheet);
      s:= '.S';                                 { return & call a worksheet }
   end;
end.
```

```
Program QERFOR(f: string) [menu];
{     This program selects one of the SPECS display formats          }
{     A moving arrow and highlighting are used as markers            }
{     The program is invoked from the menu FXGEN, option F           }

                    { **** by M.Muraszkiewicz, October 4, 1991 **** }

var     sop, lin, col, hi, wi, le            :real;
        str, fornm                           :string;

Function SELECT(stlin,stcol,high,wide,len :real;str :string): real;
    {  This function allows for selecting an option by means    }
    {  of an arrow and highlighting.                            }
    {     Input paramaters:  stlin, stcol, high, wide, len      }
    {    Output parameters: SELECT, str                         }
var sc, i, k :real;
begin
i:= 0; k:= 0; str:= '';
repeat
CURSOR(stlin+1+i,stcol+wide);
writeln('<----');                                  { draw  new arrow    }
ATTR(' ',0,26,80,1);                               { hide a cursor      }
sc:= KBDKEY(str);
if sc = 80 then i:= i+1;                           { down arrow was met }
if sc = 72 then i:= i-1;                           { up arrow was met   }
if i > high-3 then i:= 0;                          { skip to the top    }
if i < 0 then i:=high-3;                           { skip to the bottom }
if sc=72 or sc=80 then
   begin
   CLEARBOX(stlin+1+k,stcol+wide,1,5,0);           { remove old arrow   }
   CHATTR(0,stlin+1+k,stcol+1,len);                { remove old hlight  }
   CHATTR(2,stlin+1+i,stcol+1,len);                { put new highlight  }
   end;
k:= i;
until sc=28 or str='x' or str='X';                 { ENTER or X was met }
SELECT:= k;
end;

Procedure INDEX_BOX(stlin,stcol,high,wide,len :real);
{  Draws a box and fills it out with indexes names  }
begin
BOX(stlin,stcol,high,wide,1);
ATTR(' ',2,stlin+1,stcol+1,len);                   { 2 - hlight  }
CURSOR(stlin+1,stcol+1); write('General Format');
CURSOR(stlin+2,stcol+1); write('Format MAILING');
CURSOR(stlin+3,stcol+1); write('Proofreading Format');
CURSOR(stlin+high+1,stcol+11); write('X - Exit');
end;

Procedure MESSAGE;
begin
CURSOR(3,27); write('Formats of the QERE database');
CURSOR(22,1);
write('Select your format by means of arrows  ');
```

```
CURSOR(23,1); write('Type ENTER to confir you choice');
CURSOR(23,1); write('Strike X to quit');

end;
                { -------- Body of Program -------- }
begin
CLEAR;                                          { clear screen }
MESSAGE;                                        { display msgs }
CURSOR(4,3);

   { ---------- Display and pick up a format name ---------- }
   lin:= 8; col:= 22; hi:= 5; wi:= 39; le:= wi-2;
   INDEX_BOX(lin,col,hi,wi,le); CURSOR(lin-1,col+1); {draw a box + opt.}
   sop:= SELECT(lin,col,hi,wi,le,str);              { pick up by <--- }
   if str <> 'x' and str <> 'X' then
      begin                                         { X - Exit not met }
      { --------- formats --------- }
      if sop = 0 then fornm:='QERE';
      if sop = 1 then fornm:='MAIL';
      if sop = 2 then fornm:='PROOF';
      GETFMT('@':fornm);
      end;
   f:= ' ';                                          { return to menu }
end.
```

```
Program QERIND(s: string) [menu];
{     This program selects one of the QERE  Indexes for printing    }
{     A moving arrow and highlighting are used as markers           }
{     The program is invoked from the menu FXPRT, option S          }

                        { **** by M.Muraszkiewicz, Oct. 2, 1991 **** }

var     sop1, sop2, lin, col, hi, wi, le     :real;
        str, sheet                           :string;


Procedure DSPL(nrow,ncol: real; text: string);
begin
CURSOR(nrow,ncol); WRITE(text);
end;

Function SELECT(stlin,stcol,high,wide,len :real;str :string): real;
    {  This function allows for selecting an option by means   }
    {  of an arrow and highlighting.                           }
    {    Input paramaters:  stlin, stcol, high, wide, len      }
    {    Output parameters: SELECT, str                        }
var sc, i, k :real;
begin
i:= 0; k:= 0; str:= '';
repeat
CURSOR(stlin+1+i,stcol+wide);
writeln('<----');                                  { draw  new arrow    }
ATTR(' ',0,26,80,1);                               { hide a cursor      }
sc:= KBDKEY(str);
if sc = 80 then i:= i+1;                           { down arrow was met }
if sc = 72 then i:= i-1;                           { up arrow was met   }
if i > high-3 then i:= 0;                          { skip to the top    }
if i < 0 then i:=high-3;                           { skip to the bottom }
if sc=72 or sc=80 then
   begin
   CLEARBOX(stlin+1+k,stcol+wide,1,5,0);           { remove old arrow   }
   CHATTR(0,stlin+1+k,stcol+1,len);                { remove old hlight  }
   CHATTR(2,stlin+1+i,stcol+1,len);                { put new highlight  }
   end;
k:= i;
until (sc=28) or (str='x') or (str='X');           { ENTER or X was met }
SELECT:= k;
end;

Procedure INDEX_BOX1(stlin,stcol,high,wide,len :real);
{  Draws a box and fills it out with 2 options  }
begin
BOX(stlin,stcol,high,wide,1);
ATTR(' ',2,stlin+1,stcol+1,len);                   { 2 - hlight  }
DSPL(stlin+1,stcol+1,'Hits');
DSPL(stlin+2,stcol+1,'Whole database');
DSPL(stlin+high+1,stcol+4,'X - Exit');
end;
```

```
Procedure INDEX_BOX2(stlin,stcol,high,wide,len :real);
{  Draws a box and fills it out with indexes names  }
begin
BOX(stlin,stcol,high,wide,1);
ATTR(' ',2,stlin+1,stcol+1,len);                    { 2 - hlight  }
DSPL(stlin+1,stcol+1,'List of References');
DSPL(stlin+2,stcol+1,'Subject Index');
DSPL(stlin+3,stcol+1,'MAILING');
DSPL(stlin+high+1,stcol+4,'X - Exit');
end;
Procedure MESSAGE;
begin
DSPL(2,25,'Index of the database QERE');
CURSOR(22,1);
write('Select your format by means of arrows  ');
CURSOR(23,1); write('Type ENTER to confir you choice');
CURSOR(23,1); write('Strike X to quit');
end;
                { -------- Body of Program -------- }
begin
CLEAR;                                              { clear screen }
MESSAGE;                                            { display msgs }
CURSOR(5,5);
       { --------- Hits or Whole database ? ----------- }
write('Do you want to create hits or whole database index?');
lin:= 9; col:= 29; hi:= 4; wi:= 17; le:= wi-2;
s:= ' ';                                            { return to print menu }
INDEX_BOX1(lin,col,hi,wi,le);                       { draw a box + options }
sop1:= SELECT(lin,col,hi,wi,le,str);                { pick up by <----     }
CLEAR;
if (str <> 'x') and (str <> 'X') then
   begin                                                 { X - Exit not met }
   { ---------- Display and pick up ENTRE Indexes ----------- }
   MESSAGE; lin:= 8; col:= 23; hi:= 5; wi:= 35; le:= wi-2;
   INDEX_BOX2(lin,col,hi,wi,le); CURSOR(lin-1,col+1); {draw a box + opt.}
   if sop1 = 0 then write('HITS');
   if sop1 = 1 then write('WHOLE DATABASE');
   sop2:= SELECT(lin,col,hi,wi,le,str);                 { pick up by <--- }
   if (str <> 'x') and (str <> 'X') then
      begin                                             { X - Exit not met }
      { ------------------ hits ----------------- }
      if sop1 = 0 and sop2 = 0 then sheet:='fyq1h';
      if sop1 = 0 and sop2 = 1 then sheet:='fyq2h';
      if sop1 = 0 and sop2 = 2 then sheet:='fyq3h';
      { -------------- whole database ------------ }
      if sop1 = 1 and sop2 = 0 then sheet:='fyq1b';
      if sop1 = 1 and sop2 = 1 then sheet:='fyq2b';
      if sop1 = 1 and sop2 = 2 then sheet:='fyq3b';
      AUTOTYPE(sheet);
      CLEARMSG; CURSOR(22,21); write(sheet);
      s:= '.S';                                 { return & call a worksheet }
      end;
   end;
end.
```

```
Program SEMDIS(d: string) [menu];
{     This program selects one of the display formats                    }
{     A moving arrow and highlighting are used as markers                }
{     The program is invoked from the menu EXGEN, option D               }

                    { **** by M.Muraszkiewicz, Aug. 15, 1991 ***** }

var    flag, fn, le, ind, nx_r_tg, dummy          :real;
       r_num, frc, lrc, n_hit, i, l_count         :real;
       code, fornm, line, s, dum                  :string;


    { ---------------------- Body of Program ---------------------- }
begin
CLEAR;
l_count:= 1;
nx_r_tg:= 0;
flag:=0;
i:= 1;
n_hit:= SETPOS(0,0);                                { number of hits }
if (n_hit>0) then
  begin
  WHILE i<=n_hit DO
       begin
       r_num:= SETPOS(0,i);                          { record number }
       ind:= RECORD(r_num);                         { get the record }
       i:= i+1;
       if (ind<=0) then                  { was the record deleted ? }
          begin
    { ---- get type of record --- }                              { no }
          fn:= FIELDN(3,1);
          code:= FIELD(fn);                   { get the field value }

    { --------- formats --------- }
        if code='I' then fornm:='ENTRE';
        if code='P' then fornm:='PROJ';

     GETFMT('@':fornm);
    { --------- display --------- }
        frc:= FORMAT(79);
        if frc=0 then
           begin
             lrc:= NXTLINE(line);
             while lrc=0 do
               begin
                 writeln(line);
                 l_count:= l_count+1;
                 lrc:= NXTLINE(line);
                 if lrc<>0 then flag:=1;
                 if l_count>22 then
                    begin
                    l_count:=1; CURSOR(24,1); write('More...');
                    le:= KBDKEY(dum);          { looking for ESC }
                    CLEAR; CURSOR(1,1);
```

```
                    if le=1 then
                       begin
                       lrc:=1;
                       i:=n_hit+2;          { ESC met }
                       end;
                    if nx_r_tg=1 then
                       begin
                       lrc:=1;
                       i:= i-1;
                       flag:=0;
                       end;
                    end;
                end; {while}
            end {frc=0}
                else writeln('   Format error ',frc:1);
          nx_r_tg:=flag;
    end; {WHILE}
    end {d<=0};
  end;  {n_hit>0}
write('*** End of display ***'); s:= INKEY;
d:= ' ';                        { return to menu }
end.
```

```
Program SPEFOR(f: string) [menu];
{     This program selects one of the SPECS display formats          }
{     A moving arrow and highlighting are used as markers            }
{     The program is invoked from the menu EFGEN, option F           }

                    { **** by M.Muraszkiewicz, October 4, 1991 **** }

var     sop, lin, col, hi, wi, le            :real;
        str, fornm                           :string;

Function SELECT(stlin,stcol,high,wide,len :real;str :string): real;
    {  This function allows for selecting an option by means    }
    {  of an arrow and highlighting.                            }
    {    Input paramaters:  stlin, stcol, high, wide, len       }
    {    Output parameters: SELECT, str                         }
var sc, i, k :real;
begin
i:= 0; k:= 0; str:= '';
repeat
CURSOR(stlin+1+i,stcol+wide);
writeln('<----');                                { draw  new arrow    }
ATTR(' ',0,26,80,1);                             { hide a cursor      }
sc:= KBDKEY(str);
if sc = 80 then i:= i+1;                         { down arrow was met }
if sc = 72 then i:= i-1;                         { up arrow was met   }
if i > high-3 then i:= 0;                        { skip to the top    }
if i < 0 then i:=high-3;                         { skip to the bottom }
if sc=72 or sc=80 then
   begin
   CLEARBOX(stlin+1+k,stcol+wide,1,5,0);         { remove old arrow   }
   CHATTR(0,stlin+1+k,stcol+1,len);              { remove old hlight  }
   CHATTR(2,stlin+1+i,stcol+1,len);              { put new highlight  }
   end;
k:= i;
until sc=28 or str='x' or str='X';               { ENTER or X was met }
SELECT:= k;
end;

Procedure INDEX_BOX(stlin,stcol,high,wide,len :real);
{  Draws a box and fills it out with indexes names  }
begin
BOX(stlin,stcol,high,wide,1);
ATTR(' ',2,stlin+1,stcol+1,len);                 { 2 - hlight  }
CURSOR(stlin+1,stcol+1); write('General Format');
CURSOR(stlin+2,stcol+1); write('Proofreading Format');
CURSOR(stlin+high+1,stcol+11); write('X - Exit');
end;

Procedure MESSAGE;
begin
CURSOR(3,27); write('Display Formats of SPECS');
CURSOR(22,1);
write('Select your format by means of arrows  ');
CURSOR(23,1); write('Type ENTER to confir you choice');
```

```
CURSOR(23,1); write('Strike X to quit');
end;
                { -------- Body of Program -------- }
begin
CLEAR;                                          { clear screen }
MESSAGE;                                        { display msgs }
CURSOR(4,3);

   { ---------- Display and pick up a format name ---------- }
   lin:= 8; col:= 22; hi:= 4; wi:= 39; le:= wi-2;
   INDEX_BOX(lin,col,hi,wi,le); CURSOR(lin-1,col+1); {draw a box + opt.}
   sop:= SELECT(lin,col,hi,wi,le,str);              { pick up by <--- }
   if str <> 'x' and str <> 'X' then
      begin                                         { X - Exit not met }
      { --------- formats --------- }
      if sop = 0 then fornm:='SPECS';
      if sop = 1 then fornm:='PROOF';
      GETFMT('@':fornm);
      end;
   f:= ' ';                                          { return to menu }
end.
```

```
Program SPEIND(s: string) [menu];
{     This program selects one of the SPECS Indexes for printing    }
{     A moving arrow and highlighting are used as markers           }
{     The program is invoked from the menu FXPRT, option S          }

                        { **** by M.Muraszkiewicz, Oct. 2, 1991 **** }

var     sop1, sop2, lin, col, hi, wi, le     :real;
        str, sheet                           :string;


Procedure DSPL(nrow,ncol: real; text: string);
begin
CURSOR(nrow,ncol); WRITE(text);
end;

Function SELECT(stlin,stcol,high,wide,len :real;str :string): real;
    {  This function allows for selecting an option by means   }
    {  of an arrow and highlighting.                           }
    {     Input paramaters:  stlin, stcol, high, wide, len     }
    {    Output parameters: SELECT, str                        }
var sc, i, k :real;
begin
i:= 0; k:= 0; str:= '';
repeat
CURSOR(stlin+1+i,stcol+wide);
writeln('<----');                                { draw  new arrow    }
ATTR(' ',0,26,80,1);                             { hide a cursor      }
sc:= KBDKEY(str);
if sc = 80 then i:= i+1;                         { down arrow was met }
if sc = 72 then i:= i-1;                         { up arrow was met   }
if i > high-3 then i:= 0;                        { skip to the top    }
if i < 0 then i:=high-3;                         { skip to the bottom }
if sc=72 or sc=80 then
   begin
   CLEARBOX(stlin+1+k,stcol+wide,1,5,0);         { remove old arrow   }
   CHATTR(0,stlin+1+k,stcol+1,len);              { remove old hlight  }
   CHATTR(2,stlin+1+i,stcol+1,len);              { put new highlight  }
   end;
k:= i;
until (sc=28) or (str='x') or (str='X');         { ENTER or X was met }
SELECT:= k;
end;

Procedure INDEX_BOX1(stlin,stcol,high,wide,len :real);
{  Draws a box and fills it out with 2 options  }
begin
BOX(stlin,stcol,high,wide,1);
ATTR(' ',2,stlin+1,stcol+1,len);                 { 2 - hlight  }
DSPL(stlin+1,stcol+1,'Hits');
DSPL(stlin+2,stcol+1,'Whole database');
DSPL(stlin+high+1,stcol+4,'X - Exit');
end;
```

```
Procedure INDEX_BOX2(stlin,stcol,high,wide,len :real);
{  Draws a box and fills it out with indexes names  }
begin
BOX(stlin,stcol,high,wide,1);
ATTR(' ',2,stlin+1,stcol+1,len);                    { 2 - hlight  }
DSPL(stlin+1,stcol+1,'List of References');
DSPL(stlin+2,stcol+1,'Subject Index');
DSPL(stlin+high+1,stcol+4,'X - Exit');
end;

Procedure MESSAGE;
begin
DSPL(2,25,'Index of the database SPECS');
CURSOR(22,1);
write('Select your format by means of arrows  ');
CURSOR(23,1); write('Type ENTER to confir you choice');
CURSOR(23,1); write('Strike X to quit');
end;
                { -------- Body of Program -------- }
begin
CLEAR;                                              { clear screen }
MESSAGE;                                            { display msgs }
CURSOR(5,5);

       { --------- Hits or Whole database ? ----------- }
write('Do you want to create hits or whole database index?');
lin:= 9; col:= 29; hi:= 4; wi:= 17; le:= wi-2;
s:= ' ';                                            { return to print menu }
INDEX_BOX1(lin,col,hi,wi,le);                       { draw a box + options }
sop1:= SELECT(lin,col,hi,wi,le,str);                { pick up by <----     }

CLEAR;
if (str <> 'x') and (str <> 'X') then
   begin                                                  { X - Exit not met }
   { ---------- Display and pick up ENTRE Indexes ----------- }
   MESSAGE; lin:= 8; col:= 23; hi:= 4; wi:= 35; le:= wi-2;
   INDEX_BOX2(lin,col,hi,wi,le); CURSOR(lin-1,col+1); {draw a box + opt.}
   if sop1 = 0 then write('HITS');
   if sop1 = 1 then write('WHOLE DATABASE');
   sop2:= SELECT(lin,col,hi,wi,le,str);                 { pick up by <--- }
   if (str <> 'x') and (str <> 'X') then
      begin                                             { X - Exit not met }
      { ------------------ hits ----------------- }
      if sop1 = 0 and sop2 = 0 then sheet:='fyi1h';
      if sop1 = 0 and sop2 = 1 then sheet:='fyi2h';
      { ------------- whole database ------------ }
      if sop1 = 1 and sop2 = 0 then sheet:='fyi1b';
      if sop1 = 1 and sop2 = 1 then sheet:='fyi2b';
      AUTOTYPE(sheet);
      CLEARMSG; CURSOR(22,21); write(sheet);
      s:= '.S';                                 { return & cali a worksheet }
      end;
   end;
end.
```

```
Program TRANSF(q: string) [menu];
{     This program moves the specified record from one year    }
{     to the next year                                         }
{     The program is invoked from the EXE1 menu option T       }

                { **** by M.Muraszkiewicz, Aug. 18, 1991 **** }

var  fn, d, m_l, m_l_s, n_r                    :real;
     lin, col, hi, wi, le                      :real;
     t_f_tg, poz, poz1                         :real;
     r_type, wks_name, s, ss, dummy, o_y, n_y  :string;
     sc, scode, pref, stat                     :string;

Procedure MESSAGE(lin1,col1:real;text1:string;
                  lin2,col2:real;text2:string;
                  lin3,col3:real;text3:string;
                  halt:string);

{-------------------------------------------------------}
{         Displays the messages                         }
{-------------------------------------------------------}

var du   :string;

begin
cursor(lin1,col1);
writeln(text1);
cursor(lin2,col2);
writeln(text2);
cursor(lin3,col3);
write(text3);
if halt='halt' then du:= INKEY;

end; {procedure}


Function SELECT(stlin,stcol,high,wide,len :real;
                str,scode:string): real;

{-----------------------.-------------------------------}
{   This function allows for selecting an option by means  }
{   of an arrow and highlighting.                          }
{      Input paramaters:  stlin, stcol, high, wide, len    }
{      Output parameters: SELECT, str                      }
{-------------------------------------------------------}

var  sc, i, k :real;
d             :string;

begin
i:= 0; k:= 0; str:= '';
CHATTR(2,stlin+1,stcol+1,len);                { put new highlight  }
repeat
CURSOR(stlin+1+i,stcol+wide);
```

```
{writeln('<----'); }                                    { draw  new arrow    }
ATTR(' ',0,26,80,1);                                   { hide a cursor      }
sc:= KBDKEY(str);
if sc = 80 then i:= i+1;                               { down arrow was met }
if sc = 72 then i:= i-1;                               { up arrow was met   }
if i > high-3 then i:= 0;                              { skip to the top    }
if i < 0 then i:=high-3;                               { skip to the bottom }
if sc=72 or sc=80 then
   begin
   CLEARBOX(stlin+1+k,stcol+wide,1,5,0);               { remove old arrow   }
   CHATTR(0,stlin+1+k,stcol+1,len);                    { remove old hlight  }
   CHATTR(2,stlin+1+i,stcol+1,len);                    { put new highlight  }
   end;
k:= i;
until sc=28  or              { ENTER }
      sc=73  or              { PgUp  }
      sc=81  or              { PgDn  }
      str='x' or str='X';    { X was met }
scode:= ENCINT(sc,2);
SELECT:= k;
end;

Function ITEM(stlin,stcol,high,wide,len :real;
              text1,
              text2:string):real;

{-------------------------------------------------------}
{     This program selects one of the items             }
{     ITEM returns row number of the selected item      }
{     0 is returned if x or X met                       }
{-------------------------------------------------------}

var    sop, lin, col, hi, wi, le          :real;
       scode, str                         :string;

begin

{ -------- draw a box -----------}
CLEARBOX(stlin,stcol,high,wide,0);
BOX(stlin,stcol,high,wide,2);
ATTR(' ',2,stlin+1,stcol+1,len);

{ --------- write the items in the box ------- }
MESSAGE(stlin+1,stcol+1,text1,1,1,'',1,1,'','');
MESSAGE(stlin+2,stcol+1,text2,1,1,'',1,1,'','');
MESSAGE(stlin+high-1,stcol+wide-7,'X-Exit',1,1,'',1,1,'','');
MESSAGE(22,1,'Select your item by means of  , ',
        23,1,'Type ENTER to confirm you choice',
        24,1,'Strike X to quit','');

sop:= SELECT(stlin,stcol,high,wide,len,str,scode);   { pick up item }
UC(str);

ITEM:= sop+1;
```

```
if str = 'X' then ITEM:=0;
end; {function}


Function PICK_UP(pref:string;t_f_tg:real;scode:string):real;
{-------------------------------------------------------------------}
{     This program draws a window and puts there postings           }
{     from the Inverted File according to the specidication         }
{     given in the parameters section                               }
{       Input par:  pref   - prefix of the field as in FST          }
{                   t_f_tg - field tag prefixed by pref             }
{       Outpur par: MFN of the selected record                      }
{                   or 0 if there are no documents in the database  }
{                   scode = '28' if ENTER or = X                    }
{     Note! It is assumed that the Inverted File has been updated   }
{-------------------------------------------------------------------}

var    sop, lin, col, hi, wi, le, i, j                    :real;
       high, bound, full, lngth, offset, incrm            :real;
       pref_ln, d, mfn_num                                :real;
       str, fornm, du                                     :string;
       prf,  sub_first, first, text, term, term_cut       :string;

begin
PICK_UP:= 0;
scode:='';
str:='';

{ --- calculate the number of postings ---}
pref_ln:= SIZE(pref);
prf:= pref;
d:= FIND(prf);                  { some bizzard preparation }
d:= NXTPOST;
mfn_num:=POSTING('mfn');
d:= RECORD(mfn_num);
first:= FIELD(FIELDN(t_f_tg,1));
sub_first:= SUBSTR(pref:first,1,30); { first 30 characters }
UC(sub_first);
d:= FIND(sub_first);

if d = 0 then
begin
i:= 0;                                  { calculation itself }
REPEAT
  i:= i+1;
  term:=NXTTERM;
  term_cut:= SUBSTR(term,1,pref_ln);
UNTIL not(term_cut=pref);
lngth:=i;                               { number of postings }

{ ----------- Set up the window parameters ----------- }
lin:= 5;
col:= 40;
hi:=17;
```

```
wi:= 31;
le:= wi-2;
{ ------------------------------------------------- }
CLEARBOX(lin,col,hi,wi,0);
BOX(lin,col,hi,wi,2);                    { draw the window }
MESSAGE(lin+hi-1,col+wi-7,'X-Exit',1,1,'',1,1,'','');

offset:= 0;
incrm:= hi-2;
high:= hi;
bound:= VAL(ENCINT(lngth/incrm,4)) * incrm; full:= lngth - bound;
if incrm>lngth then high:= full+2;

REPEAT
   CLEARBOX(lin+1,col+1,hi-2,wi-2,0);
   d:= FIND(sub_first);
   FOR j:= 1 TO offset DO term:= NXTTERM;     { reach the offset term }

   FOR i:= 1 TO hi-2 DO
     begin
     CURSOR(lin+i,col+2);
     if offset+i <= lngth then
       begin
       if offset=0 and i=1 then term:= sub_first;
{       d:= NXTPOST;
       mfn_num:=POSTING('mfn');                  }
       term:= SUBSTR(term,pref_ln+1,30);          { remove prefix }
{       term:= term+' ===>  ':ENCINT(mfn_num,4); }
       term:= SUBSTR(term,1,wi-2);                { adjust to widow width }
       write(term);
       term:=NXTTERM;
       end;
     end;

    sop:= SELECT(lin,col,high,wi,le,str,scode); { pick up by <-- }

    if lngth<incrm then high:= lngth+2;             { i  }

    if lngth=incrm then
        begin high:= incrm+2; offset:= 0; end;      { ii  }

    if lngth>incrm then                             { iii }
       begin
       if full>0 then                               { iiia }
          begin
          high:= incrm+2;
          if scode='73' then offset:= offset-incrm;        { PgUp }
          if scode='81' then offset:= offset+incrm;        { PgDn }
          if offset+full >= lngth then high:= full+2;
          if offset > bound then offset:= offset-incrm;
          if offset<=0  then offset:= 0;
          end;

       if full=0 then                                 { iiib }
```

```
            begin
            high:= incrm+2;
            if scode='73' then offset:= offset-incrm;          { PgUp }
            if scode='81' then offset:= offset+incrm;          { PgDn }
            if offset = bound then offset:= offset-incrm;
            if offset<=0  then offset:= 0;
            end;
          end;

UNTIL str='x' or  str='X' or scode='28';            { x,X,ENTER }

UC(str);
if str='X' then scode:= str;

{---------- get mfn of the selected term -------}

        d:= FIND(sub_first);
        FOR j:= 1 TO sop+offset DO term:= NXTTERM;     { reach the offset term }
        d:= NXTPOST;
        PICK_UP:=POSTING('mfn');
end
 else
   begin
   CLEARBOX(4,1,21,70,0);    { consult col with PICK_UP col }
   MESSAGE(13,3,'There are no PROJECT records in the database',
           14,3,'or you have to update the Inverted File',
           21,3,'Type any key to continue','halt');
   end;
end;


Procedure TRANSFER(r_num_old :real;wks_name :string);

{-------------------------------------------------------------}
{    Copies the record r_num_old to the new one.              }
{    The field numbs are taken from the wks_name worksheet.   }
{-------------------------------------------------------------}

var  tagnu                                      :array[1..250] of real;
     r_num_new, n_occ, f_num, n_field, i, d     :real;
     nf, indx, l, k, tot_n_fld, f_ex_tag, j     :real;
     f_cont, line1, line12, s_n_field, seq      :string;
     str, pre                                   :string;

begin
         l:=23; k:=57;
         ATTR(' ',4,1,k,13);
         MESSAGE(1,k+1,'Please wait',1,1,'',26,1,'','');

tot_n_fld:= 0;                          { total number of fields }
pre:= 'a';
j:= 0;
f_ex_tag:= FILEXIST(path('dbn',10)|pre|wks_name);
```

```
WHILE (f_ex_tag=0) DO
begin
   ASSIGN('INP',path('dbn',10):pre:wks_name);

{ --- get the number of fields and their tags from the worksheet --- }
   FOR i:=1 TO 12 DO
     begin
     readln(INP,line12);
     if i=1 then line1:= line12;
     end;

   n_field:= VAL(SUBSTR(line1,1,3));  { number of fields on worksheet }

   FOR i:= 1 TO n_field DO
     begin
     seq:= SUBSTR(line12,(i-1)*6+1,6);
     tagnu[i+tot_n_fld]:= VAL(seq);    { field tags are in tagnu      }
     end;

   tot_n_fld:= tot_n_fld + n_field;

   j:= j+1;
   pre:= CHR(97+j);                        { produces prefix b,c,d,e,.... }
   f_ex_tag:= FILEXIST(path('dbn',10):pre:wks_name);
end; { while }

r_num_new:= NEWREC;                         { create the new record    }
UPDATE;

FOR i:= 1 TO tot_n_fld DO
  begin
  indx:= tagnu[i];
  if indx <> 0 then
     begin
     d:= RECORD(r_num_old);
     n_occ:= NOCC(indx);
     FOR 1:= 1 TO n_occ DO
       begin
       d:= RECORD(r_num_old);
       f_num:= FIELDN(indx,1);
       f_cont:= FIELD(f_num);
          { writeln('tagnu ',tagnu[i],'   old:  ',f_cont);}
       d:= RECORD(r_num_new);
       nf:= NFIELDS;
       d:= FLDADD(indx,nf+1,f_cont);
       UPDATE;
       d:= RECORD(r_num_new);
          { f_num:= FIELDN(indx,1);
            f_cont:= FIELD(f_num);
            writeln('tagnu ',tagnu[i],'   new:  ',f_cont); str:= inkey;}
       end; { for }
     end { if }
  end;   { for}
end; { procedure }
```

```
Function STATUS(m_1,t_f_tg:real;pref:string):real;

{-------------------------------------------------}
{    Checks the status (A or C)                   }
{    of the record or records which have          }
{    the same pref:t_f_tg in the Inverted file    }
{       Returns:  0 all records are A or deleted  }
{                 MFN if record is C              }
{-------------------------------------------------}
var    c_f, typ, term, stat, st          :string;
       mf_n, d                           :real;

begin

STATUS:= 0;
d:= RECORD(m_1);                        { get the record }
c_f:= FIELD(FIELDN(t_f_tg,1));          { get the field  }
st:= pref:c_f;
UC(st);
c_f:= SUBSTR(st,1,30);

d:= FIND(c_f);              { look for in Inverted File }
d:= NXTPOST;
REPEAT
mf_n:= POSTING('mfn');
d:= RECORD(mf_n);
stat:= FIELD(FIELDN(4,1));  { Archive/Current record ? }

if d=0 and stat='C' then
   begin
     fn:= FIELDN(3,1);
     typ:= FIELD(fn);   { get the record type }

     CASE typ OF
     'I': wks_name:= 'entre.fmt';
     'P': wks_name:= 'proj.fmt';
     end; { case }
     STATUS:= mf_n;
   end;

d:=NXTPOST;
UNTIL d<0;
end; {function}


Function MFN_NUMBER(r_type,scode:string):real;

{-----------------------------------------------------------}
{   Asks for the number of the record to be transferred     }
{   returns 0 if X or x met or wrong record type            }
{-----------------------------------------------------------}
```

```
var    m_1, max, d    :real;
       seq, typ       :string;
begin
m_1:=0;
scode:='';
CLEARBOX(22,1,2,80,0);
CURSOR(24,1); writeln('Type X [Enter] to leave without moving the record');
repeat
  CLEARBOX(7,1,1,80,0); CURSOR(7,7);
  write('Pls, specify MFN of the record to be moved: ');
  readln(seq); m_1:=val(seq);
  UC(seq);                                  { upper case }
  max:= MAXMFN;
  CLEARBOX(9,1,1,80,0); CURSOR(9,7);
  if (m_1>=max) then
     begin
     CLEARBOX(9,1,1,80,0); CURSOR(9,7);
     writeln('Sorry, your MFN is wrong !   Try again, pls');
     end;
until ( (m_1<max) AND (m_1>0) OR (seq='X'));
MFN_NUMBER:= m_1;

if seq<>'X' then
begin
  d:= RECORD(m_1);
  typ:= FIELD(FIELDN(3,1));   { get the record type }
  if typ<>r_type then
     begin
     MFN_NUMBER:= 0;
     CLEARBOX(8,1,17,75,0);
     MESSAGE(10,7,'Sorry, You gave a wrong record type !',
             18,7,'Type any key to return to the menu',18,41,'','halt');
     end;
end;

if seq='X' then scode:= seq;
if seq='X' then MFN_NUMBER:= 0;
end; { function }

Function INCREASE(year:string;inc:real):string;

{-----------------------------------------------}
{             Increases year by inc             }
{-----------------------------------------------}

var  n3, n4                    :real;
     s1_2, s1_3, s3, s4        :string;

begin
INCREASE:= ENCINT(VAL(year) + inc,4);
end; { function }


{ ------- Body of program --------- }
```

```
begin
CLEAR;
MESSAGE(3,7,'*** MOVING A RECORD TO THE NEXT YEAR ***',1,1,'',1,1,'','');

repeat
CURSOR(5,7); CLEARLN;
write('Have you updated the Inverted File ? [Y/N] ');
d:= KBDKEY(s); write(s); UC(s);
until ((s='Y') OR (s='N'));
if s='N' then MESSAGE(7,7,'Please, do it before moving a record !!',
                      18,7,'Type any key to return to the menu',
                      18,41,'','halt');

CURSOR(5,7); CLEARLN;

if s='Y' then
begin
{------- draw boxes an list items and select one ----}
lin:= 12; col:= 15; hi:= 4; wi:= 20; le:= wi-2;
poz:= ITEM(lin,col,hi,wi,le,'  Organisation','  Project');
if poz<>0 then
   begin                  { first box loop }
   CASE poz OF
   1: begin
       pref:='ORG=';
       t_f_tg:= 10;
       wks_name:= 'entre.fmt';
      end;
   2: begin
       pref:='PRO=';
       t_f_tg:= 200;
       wks_name:= 'proj.fmt';
       end;
   end; {case}

   lin:= 14; col:= 28; hi:= 4; wi:= 20; le:= wi-2;
   poz1:= ITEM(lin,col,hi,wi,le,'  Name/Title','  MFN');
   if poz1<>0 then
      begin
      r_type:= 'P';
      if poz=1 then r_type:= 'I';
      CASE poz1 OF        { get the number of the record to be moved }
      1: m_l:= PICK_UP(pref,t_f_tg,scode);
      2: m_l:= MFN_NUMBER(r_type,sc);
      end; {case}

      if m_l>0 and sc<>'X' and scode<>'X' then
      begin
      m_l_s:= STATUS(m_l,t_f_tg,pref);
      if m_l_s>0 then
          begin
           TRANSFER(m_l_s,wks_name);

           n_r:= MAXMFN; n_r:= n_r-1;    { new record number }
```

```
          d:= RECORD(n_r);
          r_type:= FIELD(FIELDN(3,1)); { get the record type }
{ --- increase the Financial Year (tag=5) --- }
          fn:= FIELDN(5,1);
          o_y:= FIELD(fn);
          n_y:= INCREASE(o_y,1);              { by 1 }
          d:= FLDREP(fn,n_y);
          UPDATE;
{ --- Change old record status from C to A ------- }
          d:= RECORD(m_l_s);      { get the old record }
          fn:= FIELDN(4,1);
          o_y:= FIELD(fn);
          d:= FLDREP(fn,'A');
          UPDATE;

   { --- You can update the record after moving --- }
          CLEAR; CURSOR(3,7);                          { clear the screen }
          write('*** MOVING A RECORD TO THE NEXT YEAR ***');
          MESSAGE(7,7,'The record has been moved
succesfuliy',1,1,'',1,1,'','');
          repeat
            CURSOR(9,7); CLEARLN;
            write('Do you want to update the record after moving? [Y/N] ');
            d:= KBDKEY(s); write(s); UC(s);
          until ((s='Y') OR (s='N'));

          if (s='Y') then
             begin
               CASE r_type OF
               'I': d:= WORKSHEET('entre');
               'P': d:= WORKSHEET('proj');
                end; { case }
                AUTOTYPE('M^M^M');
                d:= MODIFY(n_r);
             end; {s='Y'}
         end {d>}
         else
          begin
          wi:= 39;
          if poz1=2 then wi:= 60;
          CLEARBOX(4,1,21,wi,0);   { consult col with PICK_UP col }
          MESSAGE(15,4,'This record cannot be transferred !',
                  16,4,'It is archival or deleted',
                  22,4,'Type any key to return to the menu','halt');
          end;
       end; {sc<>X and scode<>X}
end; {poz<>0 exit by X from the second box }
end; {poz<>0 exit by X from the first box }
end; {yes, Inverted File updated}

q:= ' ';
end.
```