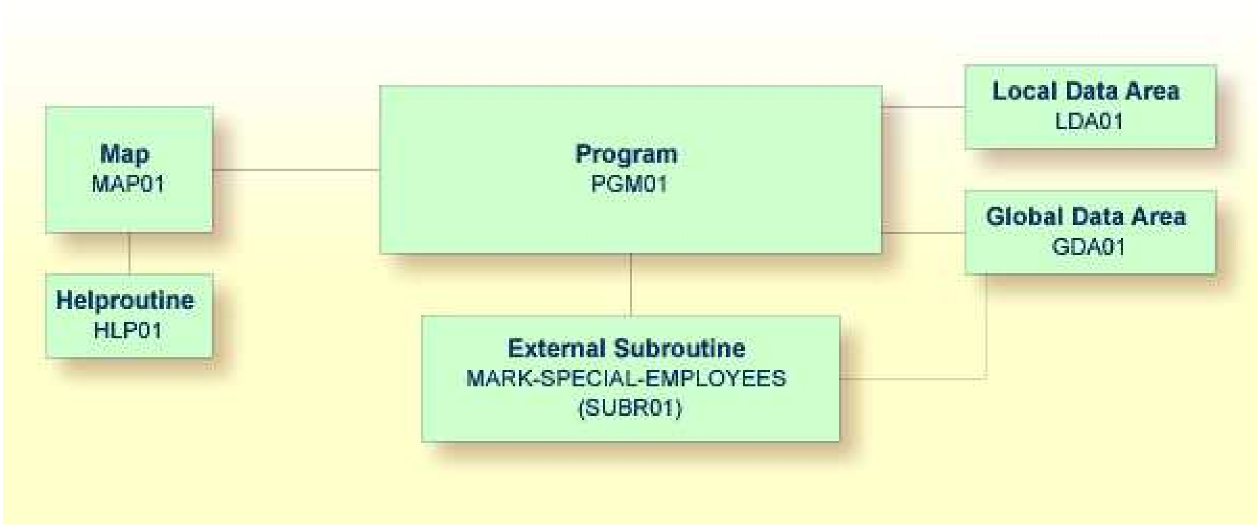
External Subroutines

Until now, the subroutine `MARK-SPECIAL-EMPLOYEES` has been defined within the program using a `DEFINE SUBROUTINE` statement. You will now define the subroutine as a separate object external to the program.

When you have completed the exercises below, your sample application will consist of the following modules:



This chapter contains the following exercises:

- Creating an External Subroutine
- Referencing the External Subroutine from Your Program

Creating an External Subroutine

Since the existing code from the program will be reused in the external subroutine, you will now save the program under a new name, change its type to subroutine and delete all lines that are not required.

The `DEFINE SUBROUTINE` statement of the external subroutine is coded in the same way as the inline subroutine in the program.

To create an external subroutine

1. Enter the following in the command line of the program editor.

```
SA SUBR01
```

The current program is saved with the new name `SUBR01`. The program is still shown in the editor.

2. Load the newly created object into the editor by entering the following command:

   ```
   E SUBR01
   ```

   The object type is still program.

3. To change the program into an external subroutine, enter the following command:

   ```
   SET TYPE S
   ```

   where "S" denotes subroutine.

   The object type which is shown in the screen changes to "Subroutine".

4. Use the line command .D to delete all lines except the following:

   ```
   DEFINE DATA
     GLOBAL USING GDA01
     LOCAL USING LDA01
   END-DEFINE
   *
   DEFINE SUBROUTINE MARK-SPECIAL-EMPLOYEES
     MOVE '*' TO #MARK
   END-SUBROUTINE
   *
   END
   ```

   It is also possible to delete a block of text. To do so, you have to proceed as follows:

   1. At the beginning of the first line of the block, enter the line command `.X`.
   2. At the beginning of the last line of the block, enter the line command `.Y`.
   3. Press ENTER.

      The block of lines to be deleted is now marked with "X" and "Y". (If you want to remove the marks, you can enter `RESET` in the command line.)
   4. To delete the marked block, enter `DX-Y` in the command line.

5. Stow the subroutine.

Referencing the External Subroutine from Your Program

The `PERFORM` statement invokes both internal and external subroutines. When an internal subroutine is not found in the program, Natural automatically tries to perform an external subroutine with the same name. Note that Natural looks for the name that has been defined in the subroutine code (that is: the subroutine name), not for the name that you have specified when saving the subroutine (that is: the Natural object name).

Now that you have defined an external subroutine, you have to remove the inline subroutine (which has the same name as the external subroutine) from your program.

To use the external subroutine in your program

1. Return to the program editor by entering the following in the command line of the editor in which the subroutine is currently shown.

   ```
   E PGM01
   ```

2. Remove the following lines:

   ```
   DEFINE SUBROUTINE MARK-SPECIAL-EMPLOYEES
     MOVE '*' TO #MARK
   END-SUBROUTINE
   ```

   Your program should now look as follows:

   ```
   DEFINE DATA
     GLOBAL USING GDA01
     LOCAL USING LDA01
   END-DEFINE
   *
   RP1. REPEAT
   *
     INPUT USING MAP 'MAP01'
   *
     IF #NAME-START = '.' THEN
       ESCAPE BOTTOM (RP1.)
     END-IF
   *
     IF #NAME-END = ' ' THEN
       MOVE #NAME-START TO #NAME-END
     END-IF
   *
     RD1. READ EMPLOYEES-VIEW BY NAME
       STARTING FROM #NAME-START
       ENDING AT #NAME-END
   *
       IF LEAVE-DUE >= 20 THEN
         PERFORM MARK-SPECIAL-EMPLOYEES
       ELSE
         RESET #MARK
       END-IF
   *
       DISPLAY NAME 3X DEPT 3X LEAVE-DUE 3X '>=20' #MARK

     END-READ
   *
     IF *COUNTER (RD1.) = 0 THEN
       REINPUT 'No employees meet your criteria.'
     END-IF
   *
   END-REPEAT
   *
   END
   ```

3. Run the program.
4. Enter "JONES" as the starting name and press ENTER.

   The resulting list should still show an asterisk for each employee who has 20 days of leave and more.
5. To return to the program editor, enter `EDIT` at the `MORE` prompt.
6. Stow the program.

To list equivalent subroutine names

1. Enter one of the following commands in the command line of the program editor:

```
LIST EXTENDED SUBROUTINE *
```

```
L EXT S *
```

The following screen appears. It lists all external subroutine objects (members) and their equivalent long names available in the current Natural library and system file.

```
12:21:09               ***** NATURAL LIST COMMAND *****             2007-03-20
User SAG                 - LIST Objects in a Library -        Library TUTORIAL

Cmd Subroutine/Class Name            Type  S/C Member    Cat Date    Cat Time
--- *________________________________ S____ --- *________ *__________ *________
__  MARK-SPECIAL-EMPLOYEES           Subro S/C SUBR01    2007-03-20  12:11:56














                                                                1 Objects found
Top of List.
Command ===>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Print Exit  Sort        --    -     +     ++          >     Canc
```

If you want to display a certain range of subroutine names, you can enter a search value followed by an asterisk (*) in the field below the header **Subroutine/Class Name** or below the header **Member**. Example:

```
Cmd Subroutine/Class Name            Type  S/C Member    Cat Date    Cat Time
--- MARK*___________________________ S____ --- *________ *__________ *________
__  MARK-SPECIAL-EMPLOYEES           Subro S/C SUBR01    2007-03-20  12:11:56
```

2. Press PF3 to return to the program editor.

You can now proceed with the next exercises: *Subprograms*.