**Level 2 Create designs and test software components**

**(7266/7267-201)**

Candidate instructions

Candidates are advised to read all instructions carefully before starting work and to

check with your assessor, if necessary, to ensure that you have fully understood

what is required.

**Time allowance: 4 hours**

**Assignment set up:** A scenario is provided for candidates in the form of a company

specification for a service they require.

This assignment is made up of four tasks:

• **Task A** - provides an outline design specification for a software component to validate

input.

• **Task B** - provides criteria that should be followed by candidates when producing them

design work.

• **Task C** - provides a specification for the software which requires functional testing.

• **Task D** - provides criteria that should be followed by candidates when producing the testing documentation.

**Scenario**

A software development company, Global Systems, develop software for use by clients. Software is being developed to create and maintain membership details for a sports club. You have been asked to design the software component that validates input. You are then required to test the whole program after development.

**Task A**

Candidates should use the following specification to fulfil the company's requirements.

In this task you are required to design the routines for validation of member records. The validation routine will be called by another routine. If a field is invalid the appropriate error message must be displayed. When every field in the screen input form has been accepted control must be passed back to the calling routine. Shown below is the screen input layout for a member record.



The fields and the validation required is shown in the following table.

|  |  |
| --- | --- |
| **Field** | **Validation** |
| Membership Number | Not empty  Modulus 11  Must be 6 digits  Numeric |
| First Name | None |
| Last Name | None |
| Address | None |
| Postcode | None |
| Sex | M or F (upper or lower  allowed) |
| Date of Birth | dd/mm/yyyy Full date check |
| Join Date | dd/mm/yyyy Full date check |
| Type of Membership | F, S, T or B (upper or lower  allowed) |
| Subscription Due Month | MMM e.g. Jan |

For an explanation of a modulus 11 validation check for the membership

number see Appendix A.

The structure chart for the validation routines is shown below.



1 Use a program design language to produce the design for the validation routines. Perform all validation as required for the design. Any assumptions you make about the design must be documented.

2 Some error codes have already been defined for the software and are shown below with their associated message.

|  |  |
| --- | --- |
| **Error Number** | **Error Message** |
| 1 | 1: Membership Number is not numeric |
| 2 | 2: Membership Number is not 6 digits |
| 3 | 3: Membership Number is not a valid modulus 11 number |
| 4 | 4: Sex must be F or M |
| 5 | 5: Membership type must be F, S, T or B |
| 6 | 6: Invalid Date of Birth |
| 7 | 7: Invalid Join Date |
| 8 | 8: File not open |
| 9 | 9: Subscription month invalid |

The error codes 10-14 are unassigned and if required can be used for extra error messages

for your routines. Document any new error messages used.

If the Membership Number is empty, no error message is to be displayed and any other data entered is not to be saved when the Save Record button is clicked.

**Task B**

*Check that you have followed the criteria below when producing the design for the validation*

*routines:*

1 The design conforms to the design specification.

2 The design uses the most appropriate data type(s).

3 The design is consistent and complete.

4 The program design language clearly shows variable names and data types, constants,

argument names and data types, return value data types and any data structures used.

5 The program design language clearly shows the beginning and end of each iteration,

selection and routine.

**Task C**

The software has now been developed and includes the routines for file creation, validation and

printing.

In this task you are required to carry out functional testing of the MEMBER software.

The structure of the software routines developed is shown in the following structure chart



The output file is created as an append file and must be opened using a suitable filename. The file

must be opened before data can be entered, validated or saved. The records are written to the file

in text format with each individual field as string data terminated with a carriage return. The file can

be opened, read and printed using a text editor (eg Notepad).

The print layout for the printed records is shown below.

Member Records Page Z9

Z9/Z9/Z9

Number First Last Address Sex Date of Join Type Subs

Name Name Birth Date Month

999999 XXXXXXXXXX XXXXXXXXXX XXXXXXXXXXXXXXXXXXXX X 99/99/9

9

99/99/

99

X XXX

XXXXXXXXXXXXXXXXXXXX

XXXXXXXXXXXXXXXXXXXX

XXXXXXXXXX

999999 XXXXXXXXXX XXXXXXXXXX XXXXXXXXXXXXXXXXXXXX X 99/99/9

9

99/99/

99

X XXX

XXXXXXXXXXXXXXXXXXXX

XXXXXXXXXXXXXXXXXXXX

XXXXXXXXXX

Note: Where 9 represents a digit, X represents an alphanumeric character and Z represents zero suppression.

Ten member records must be printed per page and then a new page thrown with a heading.

1 Prepare a test plan to carry out functional testing of the software.

2 Prepare the test data to be used with the test plan.

3 Use the test plan and test data to carry out a series of tests and record the test results in a test log.

4 Use the test log to produce a report which specifies the presence or absence of errors and makes proposals for rectifying errors and reports on the success of the test against the original software specification.

**Task D**

*Check that you have followed the criteria below when producing the testing documentation:*

1 The test plan contains a test number, date, purpose and type of test and expected outputs for stated inputs.

2 The test data tests the software execution under normal and exceptional circumstances.

3 Evidence of printed output, screen prints and file output must be cross referenced to the correct test number.

**Note**

• Candidates should produce the following for their assessor:

• Program design language algorithms for the validation routines.

• Test plan, test data and test log for the testing.

• A report on test results.

• Cross referenced evidence of testing ie screen prints, printed output and file output as

necessary to show test results.

• At the conclusion of this assignment, hand all paperwork and removable media to the test supervisor.

• Ensure that your name is on the removable media and all documentation.

• If the assignment is taken over more than one period, all removable media and paperwork must be returned to the test supervisor at the end of each sitting.

**Modulus 11 check**

A modulus 11 check is carried out as follows:

Multiply each digit in the membership number, staring at the right, by the number 1, then 2, then 3

etc.

Multiply by **6 5 4 3 2 1**

**Membership**

**number**

1 3 5 2 7 5

**Result** 6 15 20 6 14 5

The result of each multiplication is added together.

6 + 15 + 20 + 6 + 14 + 5 = 66

The result of the addition is then divided by the modulus (11).

66 divided by 11 = 6 reminder 0

If the remainder from the division is 0 the membership number is a valid modulus 11 number

otherwise the membership number is not a valid modulus 11 number.

The remainder is 0 so the membership number 135275 is a valid modulus 11 number.