UNIVERSITI SAINS MALAYSIA

First Semester Examination
Academic Session 1999/2000

September 1999

CSI502 - Problem Solving and Programming

Duration : [3 hours]

INSTRUCTION TO CANDIDATE:

- Please ensure that this examination paper contains **FOUR** questions in **NINE** printed pages before you start the examination.
- Answer **ALL** questions.
- This is an 'Open Book' examination.
- You are allowed to bring in any references into the examination hall.
- You can choose to answer either in Bahasa Malaysia or English.

ENGLISH VERSION OF THE QUESTION PAPER

1. (a) Explain in your own words, why high cohesion is important in program design. You may use appropriate example(s) to support your explanation.

[20/100]

(b) (i) An algorithm is required to read a series of exam marks (one at a time) and prints as either pass (mark >= 50) or fail (mark < 50). The series will end with a negative value. Write your solution in a flow-chart form.

[15/100]

(ii) Now suppose a new grade point (GP) system is introduced with the following conversion rules:

Mark	GP
80 - 100	4.00
70 - 79	3.67
64 - 69	3.33
58 - 63	3.00
52 - 57	2.67
46 - 51	2.33
40 - 45	2.00
36 - 39	1.67
32 - 35	1.33
28 - 31	1.00
25 - 27	0.67
0 - 24	0.00

Modify your algorithm above to incorporate this new requirement. For each valid mark entered by the user the system has to do the following:

- accumulate the mark in each category,
- increment the counter in each category

When a user enters a negative value, the system terminates and prints the following outputs:

- average mark in each category, and
- total number of students in each category
- GP for each category

For this question, answer in pseudocode form.

[25/100]

(c) A C program contains the following declarations and initial assignments:

```
int   i = 13, j = 9;
char  c = 'c', d = 'd';
float x = 0.5, y = -10;
```

Determine the output of each the following C statements. Used the values initially assigned to the variables for each statement.

(i) `printf("%d",3 * i - 2 * j % 2 * d - c);`

(ii) `printf("%d",2 * ((i/5) + (4 * (j - 3)) % (i + j - 2)));`

(iii) `printf("%d",(j > 9) ? i : j ? j : 0);`

(iv) `printf("%c",(c < d) ? c + 2 : d);`

(v) `printf("%d",x + y);`

[25/100]

(d) Illustrate the usage of preprocessor #define through example. Explain why you should use it in a C program.

[15/100]

2. (a) Study the codes below carefully. What will be printed when the following codes are executed?

```
int i, j, x=0;

for(i=4; i >= 0; i--)
    for(j=0; j < i; ++j) {
        x += i + j - 1;
        printf("%d ",x);
    }
printf("\nFinal value of x is %d\n",x);
```

[20/100]

(b) (i) Rewrite the conditional expression below to **if-else** statement.

```
result = (a > b) ? 1 : (a < b) ? -1 : 0
```

(ii) Do the following conditional expression produce the same result as the one in part 2 (b)(i) above? Justify your answer.

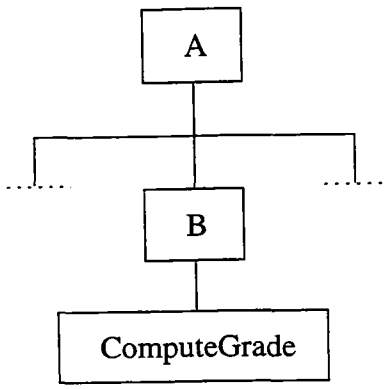
```
result = (a < b) ? -1 : (a > b)
```

[20/100]

(c) Your task is to test a function ComputeGrade which computes the exam grade (A-D) corresponding to a given mark (in the range 0 to 100 inclusive). Marks of 70 or more awarded A, from 50 to 69 inclusive a B, from 30 to 49 inclusive a C, and below 30 a D. The function specification is as follows:

Name:	ComputeGrade	
Parameter:		
IN	Mark	The raw mark (an integer)
OUT	Grade	The corresponding grade (a character)
OUT	OK	Set to true if the mark is valid (an integer)
Description:	If Mark is valid, OK return true and Grade returns the grade corresponding to Mark. If the Mark is not valid, OK returns false and Grade is undefined.	
Calling Modules:	B	

The structure chart that shows the design of the entire program is given below:



(i) Suppose a bottom up testing approach is chosen, write a 'driver' to test function ComputeGrade i.e. module B.

(ii) Identify a few test cases and hence a set of test data to test the function ComputeGrade. Prepare the test plan in the table format which include the test data, purpose (test cases) and the expected results.

[35/100]

(d) The program below calls a function named mystery:

```
#include <stdio.h>

/* function prototype */
...?...

main()
{
int num;

printf("Enter a number : ");
scanf("%d",&num);

/* function call */
...?...

printf("num is %d\n",num);
}

int mystery(int n)
{
    int s, x = 0;

    for(s=1; s <= n; s++)
        x += s * s;
    return s;
}
```

Answer the following questions:

(i) Write the function prototype of **mystery** in the above program.

(ii) Write the function call for **mystery** in the above program.

(iii) If num is entered a value of 3 at **scanf()**, what is the output of the program?

(iv) Suggest a meaningful (better) name for the function **mystery** that explain its purpose.

[25/100]

3. (a) Consider the following function code:

```
int func(int n)
{
    if (n == 0)
        return 0;
    else
        return 1 + func(n/10);
}
```

(i) What do you think is the purpose of function **func**?

(ii) Rewrite the recursive function **func** to its iterative version.

[20/100]

(b) (i) Declare a two dimensional array called **m**, and initialise its elements to the following values:

2 3 0 0
0 0 0 0
4 5 0 0
0 0 6 7

(ii) Write a function which when invoked by the statement **transpose (m,n)**, will interchange the first **n** rows and columns of the two dimensional array **m**. For example, if **n = 3** and

$$\text{initially m} = \begin{bmatrix} 2 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 4 & 5 & 0 & 0 \\ 0 & 0 & 6 & 7 \end{bmatrix} \text{ then finally m} = \begin{bmatrix} 2 & 0 & 4 & 0 \\ 3 & 0 & 5 & 0 \\ 0 & 0 & 0 & 6 \\ 0 & 0 & 0 & 7 \end{bmatrix}$$

[25/100]

(c) The function below employed a linear search method which search for the value **key** that matches the element of array **x**. If a match is found, an index of the matching element is returned, otherwise a -1 value is returned instead.

```
int linearSearch(int x[], int key, int size)
{
    int i;

    for(i=0; i < size; i++)
    if (key == x[i])
        return i;
    return -1;
}
```

(i) Explain why this method of searching is inefficient for a large array?

(ii) Modify the function above so that it will return the index of an array **x** where the last match is found, or return -1 if no match is found.

[25/100]

(d) Read the following program specification carefully.

You're required to write a fully working C program that will do the following:

- Call a function named **GetHours()**. This function should allow you to input a value into a local variable called **H**. The value of **H** is then returned and assigned to the variable **Hours** of **main()**. Validate user input before returning result to the caller function.

- Call a function named **GetRate()**. This function should allow you to input a value into a local variable called **R**. The value of **R** is then returned (after validation) and assigned to the variable **Rate** of **main()**.

- Call a function named **CalcWage()** by passing in the value of **Hour** and **Rate**. The function should use two formal parameters called **Hr** and **Rt**, to calculate the wage earned by using the formula **Hr * Rt**. This calculated value is then returned and assigned to the variable **Wages** which is defined in **main()**.

- Call a function named **CalcScale()** by passing in the value of **Wage**. **CalcScale** should use a formal parameter named **Wg** to store this value. It then determines and returns the appropriate tax scale character, based on the information found in the following table:

Wg	Tax Scale character
above 3000	'A'
1500 - 2999	'B'
below 1500	'C'

Refer to the following program skeleton to help you in writing your program:

```
main()
{
    int Hours;
    float Rate, Wage;
    char TaxScale;

    ... = GetHours();
    ... = GetRate();
    ... = CalcWage(..., ...);
    TaxScale = CalcScale(...);
    printf("Wage earned is %...?...",...);
    printf("Tax Scale is %...?...",...);
}
```

[30/100]

4. (a) Based on the following description, define using **struct** keyword the structure of book and borrower.

"A library maintains two important entities, book and borrower. A record of a book consists of the following data, the author, the publisher, call number, year of publication, edition, and its status (reserved, borrowed, available). A record of a borrower on the other hand consists of the following data, the borrower name, borrower matric number, department, and telephone number. In addition, a borrower is classified into two main groups, staff and student. For the staff group, a member is either academic or non-academic, and for the student a member is either postgraduate student or undergraduate student. In each of the categories, allowable borrowing period is to be maintained."

[25/100]

(b) Given the following structures definition:

```
struct forwarder {
        int forwarder_id;
        char address[30];
        char contact_person[20];
};

struct shipment {
        int agent_id;
        int good_code;
        char destination[50];
        struct forwarder fd;
};
```

(i) Declare a normal variable called **agentA** of type **struct shipment** and a pointer variable called **agentPtr** of type **struct forwarder**.

(ii) Use function **malloc** to allocate storage for structure **forwarder**, and assign the result of allocation i.e. the base address of newly created block of memory to variable **agentPtr**.

(iii) Assign the following values to the member of struct **forwarder** referred to by variable **agentPtr**.

```
forwarder_id <-- 12345
address <-- "Port Klang (West)"
contact_person <-- "Mr. Bean"
```

[25/100]

(c) The file **RAIN.DAT** holds up to twenty years of monthly rainfall figures of Peninsular Malaysia, as follow:

```
12  8 14 28 43 .... 23 10 {Jan, Feb, Mar, etc figures for year 1}
6 10  0 12 67 ....  6  2 {Data for year 2}
. .
. .
 8  7 12  7 23 ....  9 11 {Data for year n}
```

Write a complete C program that for each month calculate the year of maximum rainfall for that month and print the result in the form below:

Month	Year
1	3
2	7
3	10
.	.
.	.
12	15

Implying that the highest January rainfall was in year 3, the highest February rainfall was in year 7, etc.

[30/100]

(d) Explain what is meant by inheritance in object-oriented paradigm. Use appropriate example(s) to support your answer.

[20/100]

- oooOooo -