UNIVERSITY OF ESWATINI

FACULTY OF SCIENCE & ENGINEERING

DEPARTMENT OF ELECTRICAL &ELECTRONIC ENGINEERING

MAIN EXAMINATION

DECEMBER 2018

TITLE OF PAPER:	**PROGRAMMING TECHNIQUES I**
COURSE CODE:	**EEE271**
DURATION:	**3 HOURS**

INSTRUCTIONS:

1. There are five (5) questions in this paper. Answer question 1 and any other three (3) questions.
2. Each question carries equal marks.
3. Use correct notation and show all your steps clearly in any program analysis.
4. All programs should be sufficiently commented and indented for clarity.
5. Start each question in a new page.

This paper should not be opened until permission has been given by the invigilator.

This paper contains seven (7) pages including this page.

Question 1 [25 Marks]

a. What causes infinite loops in a repetition structure? How can they be prevented? [3]

b. What is the purpose of a linker in a program development environment? [2]

c. In a program, what is the line of code shown below called? What does it do for your program? [3]

```
#include "my_file.h"
```

d. By illustrating with an example show how a symbolic constant is defined. [2]

e. What is the purpose of a *break* statement in a switch structure? [2]

f. Write a program segment that gives a random double value between -3 and 11 inclusive? You can assume that the random number seed has been set and any needed definitions and initializations have been made for you. [4]

g. Explain the following concepts:

i. Call by value. [3]

ii. Call by reference. [3]

h. What are the benefits of passing arguments to a function by reference? What is a drawback of using this approach, and how can it be mitigated in C? [3]

Question 2 [25 Marks]

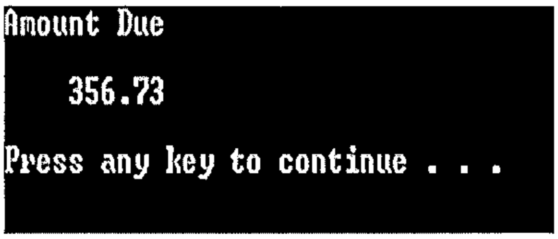
a. Assume a = 3, b = 6, c = 8, d = 9. What does the following statement print? [4]

```
printf ("Values\n%6d\n%6d\n%6d\n%6d", a != 1 && d <= 9, b == 1 || c
<= 15,c%3, b*d/c);
```

b. Given the program segment in figure Q.2b(i) , write a printf() statement that displays the exact output as shown in figure Q.2b(ii). [4]

```
        .
        .
        .
float amtDue;
scanf("%f", &amtDue);
        .
        .
```

Figure Q.2b(i)



```
Amount Due
    356.73
Press any key to continue . . .
```

Figure Q.2b(ii)

c. Consider the program segment shown in Figure Q.2c.

i. What are the values stored in array **a** after the program segment has been executed. [3]

ii. Determine the output of the program segment. [2]

```
int a[12] = {0, 1, 2, 3, 4, 5, 6, 7, 8,9,10,11};
for (i = 0, j = 1; i < 12; i += 2, j = (j++)+ i/2)
{
      a[i+1] = a[i] + j;
      printf("%2d, %2d",a[i + 1],j);
      printf("\n");
}
```

Figure Q.2c

d. What is the output after the program segment shown below has been executed. [2]

```
int i = 10, j = 20, *p, s = 0;
p = &i;
i++;
(*p)++;
s = i + j + *p;
printf("%d\n",s);
```

Figure Q.2d

e. Given the program segment shown in Figure Q.2e, briefly explain what the function ***doSomething()*** does, and write the output produced as it would appear in standard output if the given function call argument is 9. [5]

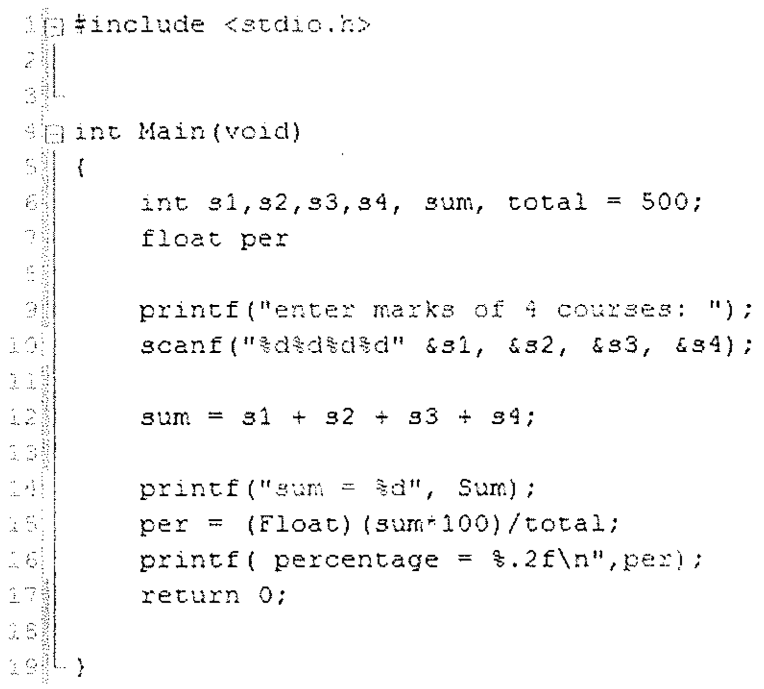
```
#include <stdio.h>

void doSomething(int value)
{
   printf("%s","==");

   if (value > 0)
         doSomething(value - 1);
   printf("\n");
}
int main()
{
   doSomething(9);

}
```

Figure Q.2e

f. Identify and describe 5 syntax errors from the C program shown in Figure 2.3. For each error specify the line at which it has been identified. [5]

Example:

Line 7: Missing semicolon at the end of statement. Every program statement must be terminated with a semicolon



```
 1  #include <stdio.h>
 2
 3
 4  int Main(void)
 5  {
 6      int s1,s2,s3,s4, sum, total = 500;
 7      float per
 8
 9      printf("enter marks of 4 courses: ");
10      scanf("%d%d%d%d" &s1, &s2, &s3, &s4);
11
12      sum = s1 + s2 + s3 + s4;
13
14      printf("sum = %d", Sum);
15      per = (Float)(sum*100)/total;
16      printf( percentage = %.2f\n",per);
17      return 0;
18
19  }
```

Figure 2.3

Question 3 [25 Marks]

a. Write a function ***reverse*** that takes as its arguments the following:
(1) an array of floating point values;
(2) an integer that tells how many floating point values are in the array.

The function must reverse the order of the values in the array. For instance, if the array passed to the function is:

0	1	2	3	4
5.8	2.6	9.0	3.4	7.1

Then when the function returns, the array will have been modified so that it is:

0	1	2	3	4
7.1	3.4	9.0	2.6	5.8

The function should not return any value.

[10]

b. For the following questions assume that reading of values is from the keyboard and output is displayed on the standard output. Further, assume that the following structure declaration has already been made.

```
struct products{
      char name[15];
      int price;
      int quantity;
      }product[10];
```

Write executable C statements with proper syntax (not a complete program), to perform each of the following tasks independently. Use only the declarations given above.

i. Enable the user to capture a list of ten products. For each product capture the name, price, and quantity. [3]

ii. Display a list of products that cost a given price. If no match is found the program should display an appropriate message. [6]

iii. Display the cheapest product in the list. [6]

Question 4 [25 Marks]

Write a function called deleteRepeats that takes in a partialy filled array and deletes all repeated letters from the array. Since a partially filled array ,requires two arguments, the function will actually have two parameters: an array parameter and a parameter of type int that gives the number of array positions used. When a letter is deleted, the remaining letters are moved forward to fill in the gap. This will create empty positions at the end of the array so that less of the array is used. The second parameter of type int must tell how many array positions are filled. This second parameter will be a call-by-reference parameter and will be changed to show how much of the array is used after the repeated letters are deleted. For example, consider the following program segment:

```
char a[10];

a[0] = 'a';

a[1] = 'b';

a[2] = 'a';

a[3] = 'c';

int size = 4;

deleteRepeats(a, size);
```

After this program is executed, the value of a[0] is 'a', the value of a[1] is 'b', the value of a[2] is 'c', and the value of size is 3. (The value of a[3] is no longer of any concern, since the partially filled array no longer uses this indexed variable.)

Note:

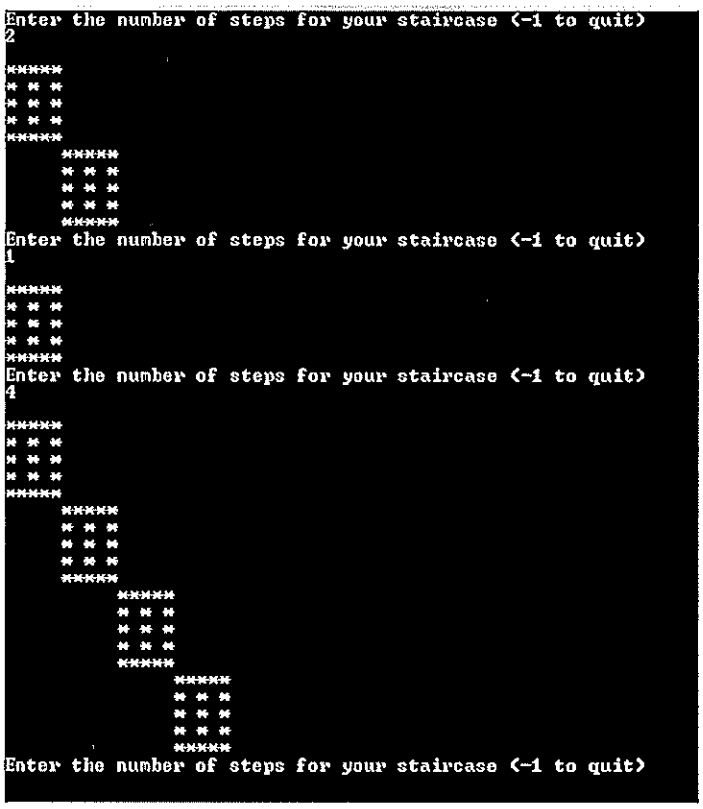
- You may assume that the partially filled array contains only lowercase letters.
- Embed your function in a suitable test program.

Your program will be graded according to the following criteria:

i.	Correctness – does the program produce the desired result?	[18]
ii.	Clarity – proper indentation of program makes it easy to read.	[3]
iii.	Sensible naming of variables – for documentation and debugging purposes	[2]
iv.	Proper use of comments – also make the program easy to read.	[2]

Question 5 [25 Marks]

Write a program that uses two functions, `step()` and `staircase()` to draw a staircase of any size. Function `staircase()` uses `step()` to draw each step. Function `step()` takes in a 5 by 5 two dimensional array that contains ones and zeros to draw the checkerboard pattern that represents the step. A ones represents an asterisk and a zero represents a space. A sample output is shown Figure Q.5.



```
Enter the number of steps for your staircase (-1 to quit)
2

*****
* * *
* * *
* * *
*****
     *****
     * * *
     * * *
     * * *
     *****
Enter the number of steps for your staircase (-1 to quit)
1

*****
* * *
* * *
* * *
*****
Enter the number of steps for your staircase (-1 to quit)
4

*****
* * *
* * *
* * *
*****
     *****
     * * *
     * * *
     * * *
     *****
          *****
          * * *
          * * *
          * * *
          *****
               *****
               * * *
               * * *
               * * *
               *****
Enter the number of steps for your staircase (-1 to quit)
```

Figure Q.5

Your program will be graded according to the following criteria:

i.	Correctness – does the program produce the desired result?	[18]
ii.	Clarity – proper indentation of program makes it easy to read.	[3]
iii.	Sensible naming of variables – for documentation and debugging purposes	[2]
iv.	Proper use of comments – also make the program easy to read.	[2]

End of Paper