00093

Index Number:

.....

Sri Lanka Institute of Information Technology

B.Sc. Eng. (Honours) Degree

Final Examination

Year 1, Semester II (2015)

EC1440 – Engineering Programming

Version A

Duration: 2 Hours

October 2015

Instructions to Candidates:

- This paper has **3 Sections**. Answer **ALL** questions, in all Sections.
- There are **15 questions** in the first Section.
  - Each question in this section carries 2 marks for a Total of 30 Marks.
- There are **5 questions** in the second Section.
  - Each question in this section carries 3 marks for a Total of 15 Marks.
- In Section 3, you have to **write 3 small programs.**
  - Two of the programs are worth 4 Marks each, and the final program is worth 7 Marks for a Total of 15 Marks.
- This paper contains 14 pages including the Cover Page, and back Page.
- No additional data is attached.
- Write your answers on this paper itself. If you need additional paper, please write you EN# on that paper and attach that paper to this paper.

page	2	3	4	5	6	7	8	9	10	total
marks										
11	12	13	14	15	16	17	18	19	20	

Section 1 (2 marks for each question)

Indicate your responses(s) by clearly making a mark in front of your selection(s).

1) Select the **INCORRECT** answer with respect to the following statement:

"functions are used in a 'C' program because:"

A) functions allow us to write re-usable code so that we can write the code once, and call it many times.
B) Use of functions allows the programmer to break the program into logical sections, so that he or she is **not** forced to write all the code into the main routine.
C) 'C' programs **cannot** be written without the use of functions.
D) functions allow for easier maintainability of code, as only a function needs to be changed, when there is a corresponding requirement change.

2) Select the **INCORRECT** statement with respect to **Non-static Variables** declared **inside a function**:

A) They are created only when the function is called.
B) They are destroyed when the function exits.
C) They can be initialized in their declaration statement.
D) They can remember values between one function call to the next.

3) The following code segment prints:

```
int i=7;

void main(void){

  printf("i=%d\n", i);
  int i=5;
  printf("i=%d\n", i);
}
```

A) i=7, i=5
B) i=5, i=7
C) i=5, i=5
D) i=7, i=7

4) The following code segment prints:

```
#include <stdio.h>

void f1(int *);

void main(void){
        int j=7;
        f1(&j);
        printf("%d\n", j);

}

void f1(int *jptr){
        printf("%d\n", *jptr);
        *jptr=9;
}
```

A) 7, 7
B) 7, 9
C) 9, 9
D) 9, 7

5) The following code segment prints:

```
#include <stdio.h>

void main(void){
   int i=4;

   printf("i=%d\n", i);
   {
      int i=5;
      printf("i=%d\n", i);
      i++;
   }
   printf("i=%d\n", i);
}
```

A) 4
   5
   4

B) 4
   5
   5

C) 4
   5
   6

D) 4
   6
   4

6) The following code segment prints:

```
#include <stdio.h>

void f1(int i);

void main(void){
   int i=4;
  f1(i);
   printf("i=%d\n", i);
}

void f1(int i){
   i++;
   printf("i=%d\n", i);
}
```

A) i=4
   i=5

B) i=4
   i=4

C) i=5
   i=5

D) i=5
   i=4

7) The following code segment prints:

```
#include <stdio.h>

void f1(int *i);

void main(void){
   int i=4;
  f1(&i);
   printf("i=%d\n", i);
}

void f1(int *i){
   (*i)++;
   printf("i=%d\n", *i);
}
```

A) i=4
   i=5

B) i=4
   i=4

C) i=5
   i=5

D) i=5
   i=4

8) The following code segment prints:

```
#include <stdio.h>

int f1(int i);

void main(void){
   int i=4;

   f1(i);
   printf("i=%d\n", i);
}

int f1(int i){
   i++;
   printf("i=%d\n", i);
   return(i);
}
```

A) i=4
   i=5

B) i=4
   i=4

C) i=5
   i=5

D) i=5
   i=4

9) The following code segment prints:

```
#include <stdio.h>

int f1(int i);

void main(void){
   int i=4;
   i=f1(i);
   printf("i=%d\n", i);
}

int f1(int i){
   i++;
   printf("i=%d\n", i);
   return(i);
}
```

A) i=4
   i=5

B) i=4
   i=4

C) i=5
   i=5

D) i=5
   i=4

10) The following code segment prints:

```
#include <stdio.h>

int f1(void);

void main(void){
   printf("i=%d\n", f1() );
   printf("i=%d\n", f1() );
}

int f1(void){
   int i=3;
   i++;
   return(i);
}
```

A) i=4
   i=5

B) i=4
   i=4

C) i=5
   i=5

D) i=5
   i=4

11) The following code segment prints:

```
#include <stdio.h>

int f1(void);

void main(void){
   printf("i=%d\n", f1() );
   printf("i=%d\n", f1() );
}

int f1(void){
   static int i=3;
   i++;
   return(i);
}
```

A) i=4
   i=5

B) i=4
   i=4

C) i=5
   i=5

D) i=5
   i=4

12) The following code segment prints:

```
#include <stdio.h>

void f1(void);

void main(void){
   f1();
   f1();
}

void f1(void){
   static int i=3;
   i++;
   printf("i=%d\n", i);
}
```

A) i=4
   i=5

B) i=4
   i=4

C) i=5
   i=5

D) i=5
   i=4

13) The following code segment prints:

```
#include <stdio.h>

void main(void){
   int a[]={11,13,15,17,19,21};
   int *ptr;

   ptr=a;

   printf("%d\n", *ptr+1);
   printf("%d\n", *(ptr+1));
}
```

A) 11
   12

B) 12
   12

C) 12
   13

D) 13
   13

14) The following code segment prints:

```
#include <stdio.h>

void main(void){
   int a[]={11,13,15,17,19,21};
   int *ptr;

   ptr=a;

   printf("%d\n", *ptr++);
   printf("%d\n", *ptr++);
}
```

A) 11
12

B) 12
12

C) 12
13

D) 13
13

15) The following code segment prints:

```
#include <stdio.h>

void main(void){
   int a[]={11,13,15,17,19,21};
   int *ptr;

   ptr=a;

   printf("%d\n", *++ptr);
   printf("%d\n", *++ptr);
}
```

A) 11
12

B) 12
12

C) 12
13

D) 13
13

Section 2 *(3 marks for each question)*

16) Consider the following code segment and select the **INCORRECT** statement from the options given below:

```
#include <stdio.h>

void main(void){
        char a[10]="array a[]";
        char b[10]="array b[]";
        char *i,*j;

        for(i=a, j=b ; (*i++ = *j++)!='\0' ;   ) ;

        printf("%s\n", a);
        printf("%s\n", b);
}
```

A) This code copies the contents of array b[] to array a[].
B) ***a*** is the starting address of array ***a[]***, ***b*** is the starting address of array ***b[]***. pointers i and j are set to ***a*** and ***b*** respectively.
C) The code works by copying *j to *i then moving i to i=i+1 and j to j=j+1
D) This code will not compile, as the third section of the for loop is empty.

17) Write a small program to copy the contents of array b[] to array a[] by filling the underlined space in the code stub below. Use array syntax to write this program.

```
#include <stdio.h>

void main(void){
        char a[10]="array a[]";
        char b[10]="nullify";

        int i;

        for(i=0 ;(_____ = _____)!='\0'; i++) ;

        printf("%s\n", a);
        printf("%s\n", b);
}
```

Questions 18 to 20 use the following code. Refer to this code when answering these questions.

Parts of the code that are relevant to each question, are reprinted next to the question.

```
/*
        cordinates. Using structs to store cordinates on and X Y plane
*/
#include <stdio.h>

struct Point{
        int x;
        int y;
};
struct Line{
        struct Point first;
        struct Point second;
};
struct Circle{
        struct Point center;
        int radius;
};

struct Point set_values(int xc, int yc);
void print_p( struct Point p);
void print_l( struct Line l);
void print_c( struct Circle c);

void main(void){
        struct Point p1={2,3},p2,p3;
        struct Line l1={p1,{5,6}};

        p2=set_values(5,2);
        p3=p2;
        print_l(l1);            //print a line

        struct Circle c1={p2, 5};
        print_c(c1);            //print a circle
}

void print_l ( struct Line l){
   printf("2 point on Line:");
   print_p(l.first);
   printf(",");
   print_p(l.second);
   printf("\n");
}
```

18) fill in the blanks in the function set_values() below, to copy x and y cordinate values to a variable of type struct Point. Return that variable back to the caller, before the function exists.

```
struct Point set_values(int xc, int yc){
        struct Point tmp;

        _____=xc;

        _____=yc;

        return( _____);
}
```

19) write a printf statement for function print_p () so that the function print_p() correctly prints a point in the format: (2,3)

```
void print_p(struct Point p){

        printf("(%d,%d)", _____ , _____ );

}
```

20) Complete writing the function print_c(), so that the function prints the radius and the coordinates of the center of a circle.

write your answer inside the function stub provided. Use already written functions when ever possible.

Your code should work with the rest of the code listed above.

```
void print_c( struct Circle c){

  printf("radius of Circle is %d units. ", _____); // print the radius
  printf(" center is :");


  _____;        // write the code to print the center of the circle.

  printf("\n");
}
```

Section 3 *(15 marks total, each program receives different marks.)*

(if you need additional space, please write the code on a separate paper, write your EN# and attach that paper to this paper)

1) Write a small program to print all even numbers from 1 to 100. You should show all the code needed for this program, including any libraries. ***(4 Marks)***