Indian Statistical Institute
Midterm Examination 2010
B.Math II Year I Semester
Computer Science I (Programming in C)
Part A (THEORY)

Time: 2 Hours **Date: October 6, 2010** **Total Marks: 30**

Question 1 is compulsory and carries 10 marks. From the remaining attempt any four questions. Each of these carries 5 marks. Marks are shown to the left of each question.

Note : if answer of any question is compiler dependent, assume it is gcc compiler in Linux.

[2] 1(a). What will be the output of the following C program? Briefly justify your answer.

```
#include<stdio.h>
void write(unsigned int x)
{
  unsigned int y;
  for (y = 100; y > 0; y /= 10)
  {
    putchar( (x / y) + '0');
    x = x % y;
  }
}

int main()
{
   unsigned int x=123;
   write(x);
return(0);
}
```

[2] 1(b). What will be the output of the following C program? Briefly justify your answer.

```
#include<stdio.h>
int main()
{
  int x=8,flag;
       if (x && !(x & (x-1)) == 0)
              flag=0;
      else
              flag=1;
printf("%d",flag);
return(0);
}
```

[2] 1(c). What will be the output of the following C program? Briefly justify your answer.

```
#include<stdio.h>
#define square(x)  (x)*(x)
#define reciprocal(x)  1 / x

int main(void)
{
int x=1, y=2, z=3, w1;
float w2;
   w1=square(++x);
   w2=(y+z)*reciprocal(y+z);

printf("w1=%d,  w2=%f",w1,w2);
return(0);
}
```

[2] 1(d). What will be the output of the following C program? Briefly justify your answer.

```
#include<stdio.h>
void auto_static()
{
int auto_var=1;
static int static_var=1;
printf("auto=%d, static=%d\n",auto_var, static_var);
auto_var++;
static_var++;
}

int main()
{
int i;
   for(i=0;i<3;i++)
   auto_static();
return(0);
}
```

[2] 1(e). What will be the output of the following C program (**string.c**) if you first **compile** the program to an executable file named **test** (**$gcc string.c -o test**) and then **run** it by typing **test "I am here"** at the **command line** (**$test "I am here"**)? Briefly justify your answer.

```
#include<stdio.h>
int main(int argc, char *argv[])
{
char *string=argv[1];
   while(*string)
   ++string;
printf("%d ",string-argv[1]);
return(0);
}
```

[2] 2(a). What is the **switch statement** in C?

[1] 2(b). What does the **break identifier** do in a **switch statement**?

[2] 2(c). Rewrite the following program using **switch** statement instead of **if-else** statement.

```
#include<stdio.h>
int main()
{
char c;
c=getchar();
if(c == '+' || c == '-' || c == '/' || c == '*')
      printf("operator\n");
else if(c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u')
          printf("vowel\n");
      else
          printf("consonant\n");
return(0);
}
```

[3] 3(a). Explain the meaning of the following three statements with reference to pointers:

i) int (*fn1)();
ii) int *fn2();
iii) int fn3(int *x);

[2] 3(b). What will be the output of the following C program? Briefly justify your answer.

```
#include<stdio.h>
void swap(int *x, int *y)
{
int temp;
temp=*x;
*x=*y;
*y=temp;
}

int main()
{
void (*fn_ptr)();
fn_ptr=swap;
int x=5, y=7;
(*fn_ptr)(&x,&y);
printf("x=%d, y=%d\n",x,y);
swap(&x,&y);
printf("x=%d, y=%d\n",x,y);
return(0);
}
```

[1] 4(a). Which of the four assignments is legal after the following declaration?
int A[10], B[20], *C;

(1) A = B; (2) B = A; (3) A = C; (4) C = A;

[1] 4(b). Which of the two is equivalent to the following declaration?
char c='Q'; char *char_ptr=&c;

(1) char c='Q'; char *char_ptr; char_ptr=&c;
(2) char c='Q'; char *char_ptr; *char_ptr=&c;

[1] 4(c). Why is *sptr='Q' **invalid** while array[0]='Q' **valid** after the following declaration?
char *sptr="BANGALORE";
char array[]="BANGALORE";

[2] 4(d). What will be the output of the following C program? Briefly justify your answer.

```
#include<stdio.h>
int main()
{
int a[][3]={10,20,30,40,50,60};
int (*ptr)[3]=a;

printf("%d, %d\n",(*ptr)[0],(*ptr)[1]);
ptr++;
printf("%d, %d\n",(*ptr)[0],(*ptr)[1]);
return(0);
}
```

[2] 5(a). What are **bit fields**?
[1] 5(b). What is the use of **bit fields** in a structure declaration?

[2] 5(c). Assume that on a certain machine an **unsigned int** variable is of size 4 bytes. Assume further that the **sizeof()** function call returns the size of its operand in bytes. Consider the following two structures:

```
struct bits1
{
unsigned int f1:10;
int word;
unsigned int f2:10;
};

struct bits2
{
unsigned int f1:10;
unsigned int f2:10;
int word;
};
```

What will **sizeof(struct bits1) and sizeof(struct bits2)** return on this machine? Briefly justify your answer.

[2] 6(a). What are the differences between **union** and **structure**?

[1] 6(b). In the following declaration, the s2 is a **pointer to character** or **character**?

```
typedef char *string;
string s1, s2;
```

[2] 6(c). What will be the output of the following C program? Briefly justify your answer.

```
#include<stdio.h>
struct address
{
char *name;
char *street;
int pin;
}ISIBC_address={"ISIBC","8th Mile Mysore Road",
560059},*add=&ISIBC_address;

int main()
{
printf("%s, %s, %d",add->name, (*add).street,
ISIBC_address.pin);
return(0);
}
```

[1] 7(a). What is **recursive function** in C?

[2] 7(b). What does the following **recursive function** return in terms of the arguments **x** and **n**?

```
int f(int x, int n)
{
int temp=1;
  if(n>0)
  {
    if(n%2==1)
      temp=temp*x;
  temp=temp*f(x*x, n/2);
  }

return(temp);
}
```

[2] 7(c). Write a **non-recursive function** equivalent to the above **recursive function**.