INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR

	Stamp / Signature of the Invigilator

EXAMINATION (Class Test 1)	SEMESTER (Autumn)

Roll Number									Section		Name	
Subject Number	C	S	1	0	0	0	1	Subject Name	*Programming and Data Structures*			
Department / Center of the Student												

Question Number	1	2	3	4	Total Marks
Marks Obtained					

Please write the answers within the boxes provided or fill up the marked blanks in questions 1 through 4. Any answer written elsewhere will not be evaluated. You can use the empty spaces for rough works.

1. Answer the following questions. If you think there will be a compilation error, or a runtime error, mention them as the output. **[5x2 = 10]**

   (a) What would be the value of x after execution of the following statements [Assume the ASCII value of ‘a’ is 65].

   ```
   int x,y = 10;
   char z = ‘a;
   x = y+z;
   ```

   75

   (b) What is the output of the following program?

   ```
   #include <stdio.h>
   int main() {
       double x = 123456789.99;
       int y = x;
       printf("%d %0.2lf", y, (double)y);
       return 0;
   }
   ```

   123456789 | **123456789.00**

   (c) What would be the output of the following code?

   ```
   #include <stdio.h>
   int main(){
       int counter=5;
       while(--counter != 0)
            printf( %d , counter);
   }
   ```

   4 3 2 1

(d) What would be the output of the following code?

```
#include <stdio.h>
int main() {
    int x = 40;
    int y;
    y = (((x==40)&&(x>40))||!x);
    printf("%d",y);
    return 0;
}
```

0

(e) What would be the output of the following C program?

```
#include <stdio.h>
int main() {
    int i,arr[10] ;
    for (i = 0; i < 10; ++i)
        arr[i] =0;
    for (i = 1; i < 5; ++i)
        arr[i] += arr[i -1] + i;
    printf("%d",arr[5]);
    return 0;
}
```

0

2. (a) Consider a C program that takes a character input. If the input character is an uppercase character between 'A' and 'Z', then the program prints the character in lowercase; otherwise, it simply prints the input character. Fill up the missing code below to complete this program. **[2+1 = 3]**

```
#include<stdio.h>

int main() {
    char c;
    printf("\nEnter the character:");
    scanf(" %c",&c);

    if((c >= 'A')&&(c <= 'Z')) {

        c = c - 'A' + 'a';
    }
    printf("\nThe output character is %c",c);
}
```

(b) Consider the following C code. What will be the output of the code if the input is "`Hello 123`"(without the quotes) followed by an enter. **[0.5 + 0.5 + 1 = 2]**

```
#include<stdio.h>
int main() {
    char c; int ndigit=0, nspace=0, nother=0;
    while((c=getchar())!='\n'){
        switch(c){
            case '0': case '1': case '2': case '3': case '4':
            case '5': case '6': case '7': case '8': case '9':
                ndigit++; break;
            case ' ': case '\t': case '\n':
                nspace++;
            default:
                nother++;
        }
     }
     printf("%d %d %d",ndigit,nother,nspace);
     return 0;
}
```

3 6 1

3. A number is *Armstrong number* if the sum of its digits raised to the **third power is equal to the number itself**. For example, 371 is an Armstrong number, since $3^3 + 7^3 + 1^3 = 371$. In the following program user is supposed to enter a limit and the program prints all the Armstrong numbers from 1 to the user specified limit. Complete the program. . **[0.5+1+0.5+0.5+1+0.5+1 = 5]**

```
#include <stdio.h>
int main() {
    int number=1, originalNumber, indicator, result = 0, limit;

    printf("Enter the limit: "); /* Read the limit from the user*/
    scanf("%d", &limit);

    while(number <= limit) {
        originalNumber = number;  result = 0;
        /* Loop for updating the result */

        while (originalNumber != 0) {

             indicator = originalNumber%10;

             result +=indicator * indicator * indicator;

            originalNumber = originalNumber/10;
         }

         if(result == number)
             printf("%d is an Armstrong number.",number);

        number++;
    }
    return 0;
}
```

4. The following C program is supposed to compute even parity check bit (pc), for the given message word $M = m_0m_1m_2...m_{n-1}$. The message and parity check bits are binary (0 or 1). The parity check bit is generated based on the number of ones present in message. **If the number of ones in the message is odd, then parity check bit is 1 otherwise it is zero**. Assume the message is loaded into an array such that each array element carries one message bit ($m[0] = m_0$, $m[1] = m_1$, ...), and size of the message is specified through user input via keyboard. After generating the parity check bit from the message, the code word will be generated by inserting the parity check bit at the beginning of the array (codeword $C = pcm_0m_1m_2..m_{n-1}$). A sample input and corresponding output are given below.

```
Enter the size of message =
6
Enter the message bits
1
0
1
0
1
1
Codeword = 0101011
```

Complete the code by filling the blanks. **[0.5x10 = 5]**

```
#include <stdio.h>
int main() {
   int num_1 = 0, n, i, m[20], pc;
   printf("Enter the size of message = \n");
   scanf("%d",&n);
   printf("Enter the message bits \n");
   /* Entering message word & counting number of ones in message*/
   for(i=0;i<n;i++) {

        scanf("%d",&m[i]);

        if(m[i] == 1) num_1++;
    }
    /* Generation of parity check bit */

    if(!(num_1%2)) pc = 0;

    else pc = 1;
    /* insertion of parity check bit */

    for(i=n-1; i >=0 ; i--)

        m[i+1] = m[i];

    m[0] = pc;
    printf("Codeword = ");
    for(i=0; i<=n; i++)
        printf("%d",m[i]);       /* printing the code word */
    return 0;
}
```