INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR

Instructions:

Write answers in the space provided in the question paper itself.

Do your rough work on the space provided at in the question paper.

Write your roll number at the space provided in every sheet of the question paper.

Name:	
Roll No:	
Section:	

---------------------------- DO NOT WRITE ANYTHING BELOW THIS LINE --------------------------------

MARKS:

Q1 (15)	**Q2 (15)**	**Q3 (15)**	**Q4 (15)**	**Total (60)**

1. Answer the following questions. [5 × 3 = 15 marks]

    a) What will be the contents of array **x** after the following code segment is executed?

```
int x[]={1,2,3,4,5,6,7};
int i,j;
i=4; j=3;
x[i] = i++;
x[--j] = --i + 4;
```

ANS 1(a):

      x[] = {1, 2, 8, 4, 4, 6, 7}

    b) (i) Consider the following declaration:

```
int i=8, j=5, k=3;
```

    Determine the value for the following expression.

```
++i + --j * k++
```

ANS 1b(i): **21**

    (ii) Consider the following if-else statement structure:

```
if e1 if e2 then s1 else s2 else if e3
then s3 else s4
```

where e1,e2,e3 are logical expressions, and s1,s2,s3,s4 are statement blocks. Indicate which else statement corresponds to which logical expression by rewriting the entire statement with appropriate curly braces.

ANS 1b(ii):

```
if e1 then { if e2
                then s1
                else s2
          }
      else { if e3
                then s3
                else s4
          }
```

c) What is printed by the following program?

```c
#include <stdio.h>
int x = 5;

void f(int b, int x)
{
    static int a=0;
    if (b == 0) {printf("\n"); return;}
    a++; x+=5; printf("%d ", x);
    f(b-1, x); printf("%d ", a);
}

int main( )
{
    int a = 5;
    f(a, x);  printf("\na=%d, x=%d\n", a, x);
    return 0;
}
```

ANS 1(c):

```
10 15 20 25 30
5 5 5 5 5
a=5, x=5
```

d) What is printed by the following code segment?

```c
int i;
for (i=12; i; i-=2) putchar("IIT Kharagpur" [i]);
putchar('\n');
```

ANS 1(d):

```
rpaaKT
```

e) What is printed by the following code segment?

```
int a[]={1,2,3,4,5,6}, *iP=a+4;
printf("%d\n", iP[1]);
printf("%d\n", a[iP[-2]]);
```

ANS 1(e):

6
4

-- END OF QUESTION 1 --

REST OF THIS PAGE MAY BE USED FOR ROUGH WORK

2. This question has three parts. [7 + 5 + 3 = 15 marks]
   a) Write a C program to compute the following series summation correct up to five decimal places.

$$S = x - \frac{x^2}{2!} + \frac{x^3}{3!} - \frac{x^4}{4!} + \cdots$$

   where $0 < x < 1$.

ANS 2(a):

```c
#include <stdio.h>
main()
{
   float sum, term, x;
   int n;

   printf ("\nEnter value of x: ");
   scanf ("%f", &x);

   n = 1;    sum = 0.0;
   term = x;

   while (fabs(term) > 0.00001)
   {
      sum = sum + term;
      n++;
      term = -term * x / n;
   }

   printf ("\nSum of series: %f", sum);
}
```

   b) Write a C program to read an integer N, and generate the following pattern of stars for N lines, using nested loops.

```
*
* *
* * *
* * * *
* * * * *
.......
```

ANS 2(b):

```c
#include <stdio.h>
main()
{
   int N, i, j;
   printf ("\nEnter value of N: ");
   scanf ("%d", &N);

   for (i=0; i<N; i++)
   {
      printf ("\n");
      for (j=0; j<=i; j++)
         printf ("* ");
   }
}
```

c) What will be the output of the following program?

```c
#include <stdio.h>
int main()
{
   int i=0;
   for ( ; i<20; i++)  {
      switch (i)  {
         case 0:  i+=5;
         case 1:  i+=2;
         case 5:  i+=5;
         default: i+=4;
      }
      printf ("%d  ", i);
   }
   putchar('\n'); return 0;
}
```

ANS 2(c):

```
16 21
```

3. Write a C program in the following way. The function **int main()** reads a positive integer *n*, $1 \le n \le 1000$, and then it reads *n* positive integers in a one-dimensional array of type **int**. It also prints the input data. It calls the function:

```
void copf(int data[], int n, int cpf[])
```

where the input data is passed through the first parameter. The second parameter is the number of data *n*.

Values in the array corresponding to the third parameter is computed by the function **copf( )**. On execution of this function, **cpf[i]** will contain the number of distinct prime factors of **data[i]**, for all *i*, $0 \le i < n$. As an example, if **data[2]** is $140 = 2^2 \times 5 \times 7$, then **cpf[2]** will contain 3.

Finally **main()** prints the content of the array which is passed as the third parameter to the function **copf()**.

The function **int pc(int k)** returns the count of distinct prime factors of its argument *k*. For example, if *k* is 140, it returns 3. The function **copf()** calls the function **int pc(int k)** with every element of **data[i]** to get the corresponding count of distinct prime factors.

a) Write the function **int main()**.

b) Write the function
   ```
   void copf(int data[], int n, int cpf[]).
   ```

c) Write C expressions corresponding to $\beta_1, \beta_2, \beta_3, \beta_4, \beta_5$ in the function **pc()**
   ```
   int pc(int k){
       int num, β₁ ;
       for(β₂;  β₃ ; ++num){
           if (k%num == 0) {
               ++ count;
               while (β₄) β₅ ;
           }
       }
       return count;
   }
   ```

[$3 \times 5 = 15$ marks]

ANS 3(a):

```c
#include <stdio.h>
#define SIZE 1000
void copf(int [], int, int []);
int pc(int);
int main()
{
    int n, data[SIZE], i, cpf[SIZE];

    printf("Enter a +ve integer: ");
    scanf("%d", &n);
    printf("Enter %d +ve integers:\n", n);
    for(i=0; i<n; ++i) scanf("%d", &data[i]);
    printf("Input data:\n");
    for(i=0; i<n; ++i) printf("%d ", data[i]);
    putchar('\n');
    copf(data, n, cpf);
    printf("Corresponding prime counts are:\n");
    for(i=0; i<n; ++i) printf("%d ", cpf[i]);
    putchar('\n');

    return 0;
}
```

ANS 3(b):

```c
void copf(int data[], int n, int cpf[]){
    int i;

    for (i=0; i<n; ++i)
        cpf[i] = pc(data[i]);
    return ;
}
```

ANS 3(c):

β_1: **count = 0** β_2: **num = 2**

β_3: **k != 1 *OR* num <= k** β_4: **k % num == 0**

β_5: **k /= num**

4.

a) A country has coins of denomination 3, 5 and 10 respectively. The following function returns -1 if it is not possible to pay a value of k using these coins. Otherwise it returns the ***minimum*** number of coins needed to make the payment.

For example, **canchange(7)** will return -1. On the other hand, **canchange(14)** will return 4 because 14 can be paid as 3+3+3+5 and there is no other way to pay with fewer coins.

Write C expressions corresponding to δ_1, δ_2, δ_3, δ_4, δ_5, δ_6, δ_7 in the following function.

```
int canchange(int k)
{
   int a= -1;
   if (k==0) return 0;
   if ( δ₁ ) return 1;
   if (k < 3) δ₂ ;

   a = canchange( δ₃ ); if (a > 0) return δ₄ ;
   a = canchange(k - 5); if (a > 0) return δ₅ ;
   a = canchange( δ₆ ); if (a > 0) return δ₇ ;
   return -1;
}
```

b) Modify the function of part (a) to write a function to print the change. For example, if we call the function **printchange(14)** it should print 3+3+3+5. The function prototype is:

```
int printchange(int k)
```

[7 +8 = 15 marks]

ANS 4(a):

δ_1:   **(k ==3) || (k == 5) || (k == 10)**

δ_2: **return -1** δ_3: **k - 10**

δ_4: **a+1** δ_5: **a+1**

δ_6: **k - 3** δ_7: **a+1**

ANS 4(b):

```
int printchange(int k)
 {
        int a= -1;

        if (k==0) return 0;

        if ((k ==3)||(k == 5)||(k == 10))
                    { printf("%d", k); return 1; }

        if (k < 3) return -1;

        a = printchange(k -10); if (a > 0)
                    { printf("+10"); return a+1; }

        a = printchange(k - 5); if (a > 0)
                    { printf("+5"); return a+1; }

        a = printchange(k-3); if (a > 0)
                    { printf("+3"); return a+1; }
        return -1;
 }
```