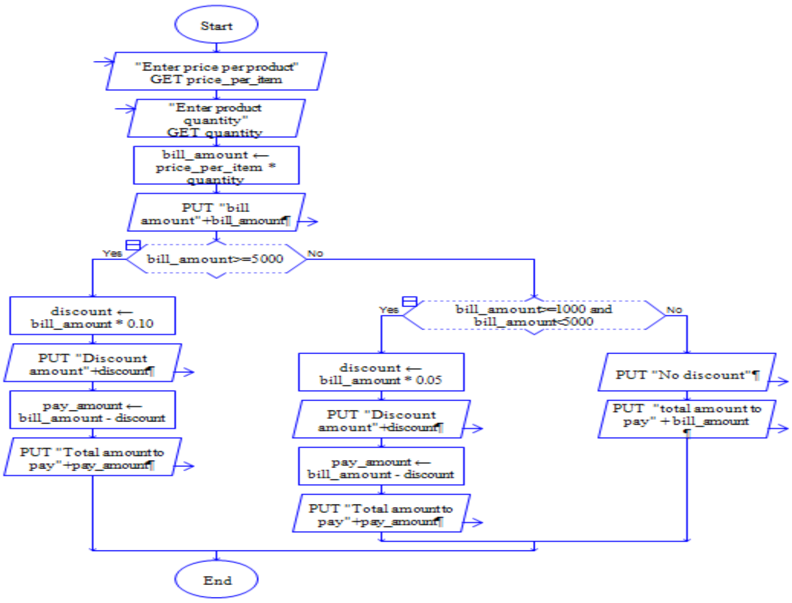
Question _scheme	**Marks**	**CO**	**Blooms level**
Q1A. (i)A compiler is a program that converts the entire source code of a programming language into executable machine code for a CPU. An interpreter takes a source program and runs it line by line, translating each line as it comes to it (ii) The compiler takes a large amount of time to analyze the entire source code but the overall execution time of the program is comparatively faster. An interpreter takes less amount of time to analyze the source code but the overall execution time of the program is slower. (iii) The compiler generates the error message only after scanning the whole program, so debugging is comparatively hard as the error can be present anywhere in the program. Its Debugging is easier as it continues translating the program until the error is met. **[1M for each]**	**3**	1	Understanding (2)
Q1B. **Typical C program development environment** ➢C programs typically go through six phases to be executed. These are: edit, preprocess, compile, link, load and execute ➢ Phase 1 : creating a program ➢Phases 2 and 3: Preprocessing and Compiling a C Program ➢Phase 4: Linking ➢Phase 5: Loading ➢Phase 6: Execution **[0.5M for explanation of each]**	3	1	Understanding(2)
Q1C. i)  Start "Enter price per product" GET price_per_item "Enter product quantity" GET quantity bill_amount ← price_per_item * quantity PUT "bill amount"+bill_amount¶ Yes bill_amount>=5000 No discount ← bill_amount * 0.10 PUT "Discount amount"+discount¶ pay_amount ← bill_amount - discount PUT "Total amount to pay"+pay_amount¶ Yes bill_amount>=1000 and bill_amount<5000 No discount ← bill_amount * 0.05 PUT "Discount amount"+discount¶ pay_amount ← bill_amount - discount PUT "Total amount to pay"+pay_amount¶ PUT "No discount"¶ PUT "total amount to pay" + bill_amount ¶ End **[1M - I/O; 2M - logic]**	4	1	Apply(3)

ii) Implicit type conversion: The compiler provides implicit type conversions when operands are of different data types. Explicit type conversion is done by the user by using (type) operator. Before the conversion is performed, a runtime check is done to see if the destination type can hold the source value. int a,c; float b; c = (int) a + b **[0.5M for each]**			

Question_scheme	**Marks**	**CO**	**Blooms level**

Q2A.
#include<stdio.h>
void main()
{
int hardness, ts, grade;
float carbon;
printf("Enter the values of hardness, tensile strength and carbon content in the steel:");
scanf("%d %d %f", &hardness, &ts, &carbon);
if ((hardness>50) && (carbon<0.7) && (ts>5600))
printf("Grade 10");
else if ((hardness>50) && (carbon<0.7))
printf("Grade 9");
else if ((carbon<0.7) && (ts>5600))
printf("Grade 8");
else if ((hardness>50) && (ts>5600))
printf("Grade 7");
else if ((hardness>50) || (carbon<0.7) || (ts>5600))
printf("Grade 6");
else
printf("Grade 5");
}
[I/O - 2M; Logic - 3M]
	5	2	Analysis (4)

Q2B.
* Determine overtime pay of 10 employees.*/
# include <stdio.h> 	
 	
int main( )
{
float otpay ;
int hour, i = 1 ;
while (i <= 10) /* Loop for 10 employees */
{
printf ("\nEnter no. of hours worked: ") ;
scanf ("%d", &hour) ;
if (hour >= 40)
{
otpay = (hour - 40) * 12 ;
printf (“No of hours worked = %d \n
Overtime pay = Rs. %f\n", hour, otpay) ;
}
else
{
otpay = 0 ;
printf (“No of hours worked (%d) is less than
40 Hrs.\nHence no overtime pay\n", hour);
}
i++;
}
return 0;
}
 	
[I/O - 1M; Logic - 2M]
	3	2	Analysis(4)

Q2C. /* Determine character case using conditional operators */ # include <stdio.h> int main( ) { char ch ; printf ("Enter character") ; scanf ("%c", &ch) ; ch >= 97 && ch <= 122 ? printf ("Character entered is lower case\n") : printf ("Character entered is not lower case\n") ; return 0; } **[Use of conditional Operators (Syntax and its placement): 1M Logic: 1M]**	2	2	Apply (3)

Question _scheme	**Marks**	**CO**	**Blooms level**

Q3A.
 #include <stdio.h>
#include <string.h>
 	
int main (void) {
   char string[50];
   gets(string);
   char temp;
 	
   int i, j;
   int n = strlen(string);
 	
   printf("String before sorting - %s \n", string);
 	
   for (i = 0; i < n-1; i++) {
      for (j = i+1; j < n; j++) {
         if (string[i] > string[j]) {
            temp = string[i];
            string[i] = string[j];
            string[j] = temp;
         }
      }
   }
 	
   printf("String after sorting - %s \n", string);
   return 0;
}
 	
    [I/O - 1M; Logic - 2M]
	3	3	Apply (3)

Q3B.
#include<stdio.h>
void main()
{ int i,j,flag,N,a[100][100];
printf("enter the value of N");
scanf("%d",&N);
for(i=0;i<N;i++)
 {for(j=0;j<N;j++)
{printf("enter element");
scanf("%d",&a[i][j]);
}
}
flag=0;
for(i=0;i<3;i++)
{ for(j=0;j<3;j++)
{ if(i>j && array[i][j]==0)
flag = 1;
}
}
if(flag==1)
    printf("upper traingular");
else
    printf("not upper triangular");
}   **[I/O - 1.5M; Logic - 1.5M]**
	3	3	Apply (3)

Q3C.	4	3	Apply (3)

```
 #include <stdio.h>
int main(){
          int a[100],n,i,j;
          printf("Array size: ");
      scanf("%d",&n);
      printf("Elements: ");

     for(i=0;i<n;i++)
   {
      scanf("%d",&a[i]);
   }
          for (int i = 0; i < n; i++)
          {
          for (int j = 0; j < n; j++)
          {
                    if (a[j] > a[i])
                    {
                              int tmp = a[i];
                              a[i] = a[j];
                              a[j] = tmp;
                    }
          }
          }
          printf("\n\nAscending : ");
          for (int i = 0; i < n; i++)
          {
          printf(" %d ", a[i]);
          }
          for (int i = 0; i < n; i++)
          {
          for (int j = 0; j < n; j++)
          {
                    if (a[j] < a[i])
                    {
                              int tmp = a[i];
                              a[i] = a[j];
                              a[j] = tmp;
                    }
          }
          }
          printf("\n\nDescending : ");
          for (int i = 0; i < n; i++)
          {
          printf(" %d ", a[i]);
          }

          return 0;
getch();
}
```

[I/O – 0.5M for each; Logic - 1.5M for each]

Question _scheme	**Marks**	**CO**	**Blooms level**
Q4A.	**3**	4	Apply (3)

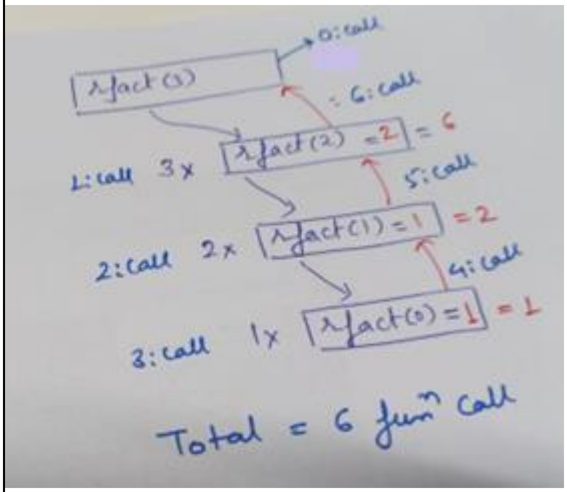
```
#include <stdio.h>
int checkPerfect(int n1);
void PerfectNumbers(int stLimit, int enLimit);
int main()
{
   int stLimit, enLimit;
          printf("\n\n Function : perfect numbers in a given range :\n");
          printf("--------------------------------------------------\n");
   printf(" Input lowest search limit of perfect numbers : ");
   scanf("%d", &stLimit);
   printf(" Input highest search limit of  perfect numbers : ");
   scanf("%d", &enLimit);

   printf("\n The perfect numbers between %d to %d are : \n",
          stLimit, enLimit);
   PerfectNumbers(stLimit, enLimit);
   printf("\n\n");
   return 0;
}
int checkPerfect(int n1)
{
   int i, sum;

   sum = 0;
   for(i=1; i<n1; i++)
   {
      if(n1 % i == 0)
      {
         sum += i;
      }
   }
if(sum == n1)
      return 1;
   else
      return 0;
}
void PerfectNumbers(int stLimit, int enLimit)
{
   while(stLimit <= enLimit)
   {
      if(checkPerfect(stLimit))
      {
         printf(" %d  ", stLimit);
      }
      stLimit++;
   }
}
```

[I/O – 1M; Logic - 2M]

Q4B. i)	5	4	Apply (3)



[2M]

ii)

```
#include<stdio.h>
long convertBinary(int);
int main()
{
   long biNo;
   int decNo;

         printf("\n\n Recursion : Convert decimal number to binary
         :\n");
         printf("---------------------------------------------------\n");

   printf(" Input any decimal number : ");
   scanf("%d",&decNo);

   biNo = convertBinary(decNo);//call the function convertBinary
   printf(" The Binary value of decimal no. %d is :
         %ld\n\n",decNo,biNo);
   return 0;
}
long convertBinary(int decNo)
{
   static long biNo,r,fctor = 1;

   if(decNo != 0)
   {
      r = decNo % 2;
      biNo = biNo + r * fctor;
      fctor = fctor * 10;
      convertBinary(decNo / 2);//calling the function convertBinary
         itself recursively
   }
   return biNo;
}
```

[I/O – 1M; Logic - 2M]

Q4C. n = 25 The output will be 1 1 0 0 1. **[2M]**	2	4	Apply (3)
Question _scheme	Marks	CO	Blooms level

Q5A.
 	
#include <stdio.h>
 	
int main()
{
int a[5],sum=0,n,*ptr, i;
printf("Enter the limit");
scanf("%d", &n);
printf("Enter the elements");
for(i=0;i<n;i++)
scanf("%d", &a[i]);
ptr=a;
for (i = 0; i < 5; i ++) {
sum += *(ptr + i);
}
printf("Sum of all array elements = %d\n", sum);
return 0;
}
 	
[Scheme: Reading n and array - 1M; sum logic using pointer - 1.5M; Output-0.5M]
	3	5	3

Q5B.
#include<stdio.h>
 	
struct student{
int marks;
char name[30];
}st[10]; **-- 1M**
 	
void find_names(struct student st[],int n)
 	
{   int i;
float total=0,avgmarks;
for(i=0;i<n;i++)
{
total = total + st[i].marks;
} **-- 1M**
avgmarks=total/n;
printf("\nAverage marks = %.2f",avgmarks);
for(i=0;i<n;i++) {
if(st[i].marks<avgmarks) {
printf("\n student %d name %s marks = %d below average",i+1,st[i].name,st[i].marks);
}
}
} **-- 1M**
	4	5	3

void main()
{
int i,n;
printf("\nEnter the number of students in class:");
scanf("%d",&n);
for(i=0;i<n;i++)
{
printf("\nEnter student %d marks :",i+1);
scanf("%d",&st[i].marks);
printf("\nEnter student %d name :",i+1);
scanf("%s",st[i].name);
}
find_names(st,n);
} **-------------------------------------- 1M**			

Q5C. a) Financial Crime: This would include cheating, credit card frauds, money laundering etc. b) Online Gambling: There are millions of websites; all hosted on servers abroad, that offer online gambling. In fact, it is believed that many of these websites are actually fronts for money laundering. c) Intellectual Property Crimes: These include software piracy, copyright infringement, trademarks violations, theft of computer source code etc. d) Email spoofing: A spoofed email is one that appears to originate from one source but actually has been sent from another source e) Cyber defamation: This occurs when defamation takes place with the help of computers and / or the Internet. Example: Someone publishes defamatory matter about someone on a website or sends e-mails containing defamatory information to all of that person's contacts. f) Cyber stalking: Cyber stalking involves following a person's movements across the Internet by posting messages (sometimes threatening) on the bulletin boards frequented by the victim, entering the chat-rooms frequented by the victim, constantly bombarding the victim with emails etc. **[1M for each crime]**	3	5	2