Ex.No : 1.1 (a) Date :	**SWAPING OF TWO NUMBERS**

AIM:

To Swap Two Numbers Using Pointers and functions.

PSEUDOCODE:

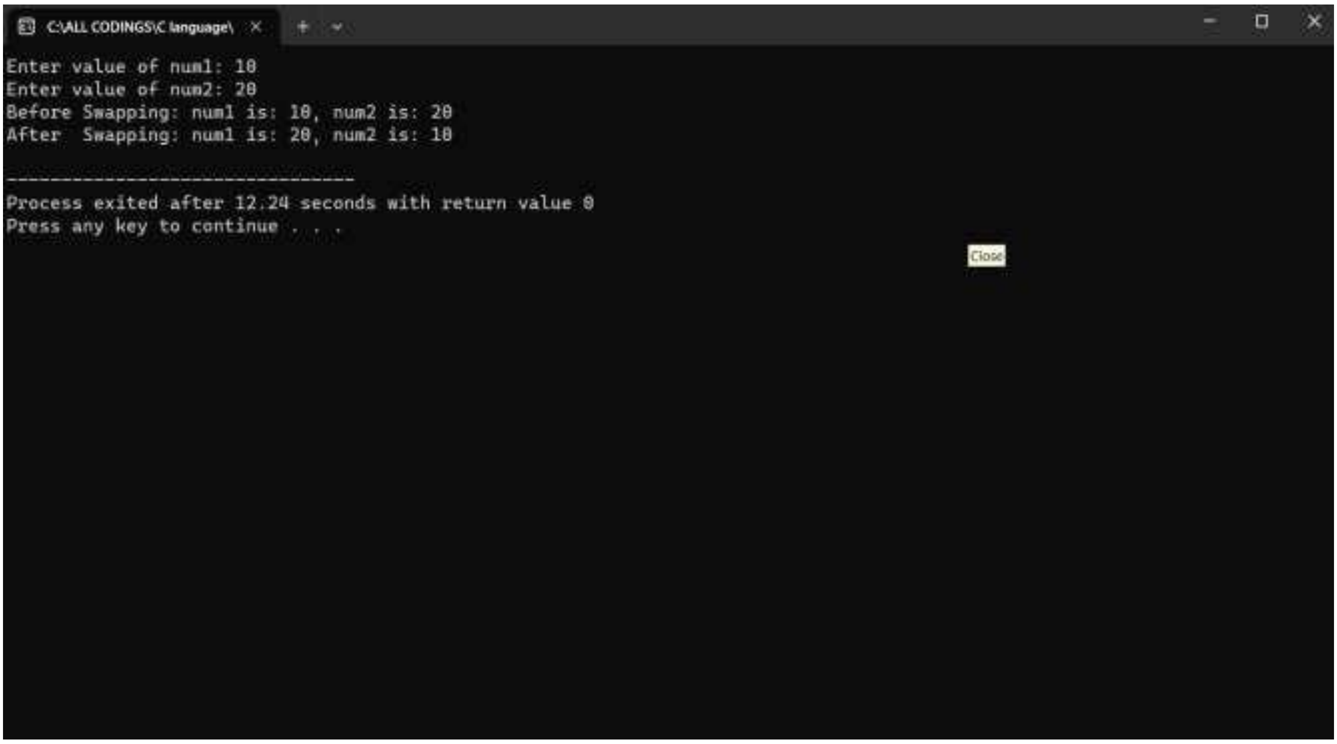
```
BEGIN
FUNCTION swap
DECLARE t
ASSIGN t=*x , *x=*y ,*y=t
FUNCTION END
DECLARE num1,num2
GET num1 ,num2
CALL FUNCTION swap
PRINT num1 and num2
END
```

SOURCE CODE :

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

```
int main()
{
    int num1,num2;

    printf("Enter value of num1: ");
    scanf("%d",&num1);
    printf("Enter value of num2: ");
    scanf("%d",&num2);

    printf("Before Swapping: num1 is: %d, num2 is: %d\n",num1,num2);
    swap(&num1,&num2);
    printf("After  Swapping: num1 is: %d, num2 is: %d\n",num1,num2);

    return 0;
}
```

OUTPUT:



```
Enter value of num1: 10
Enter value of num2: 20
Before Swapping: num1 is: 10, num2 is: 20
After  Swapping: num1 is: 20, num2 is: 10

--------------------------------
Process exited after 12.24 seconds with return value 0
Press any key to continue . . .
```

RESULT:

Thus the program to swap two numbers using pointers is executed. Successfully and the output is verified.

717822F124

Ex.No : 1.1 (b)

Date :

TO CALCULATE THE POOJA’S ACCOUNT BALANCE

AIM:

To Write C-Program to calculate Pooja’s account balance after an attemped transaction.

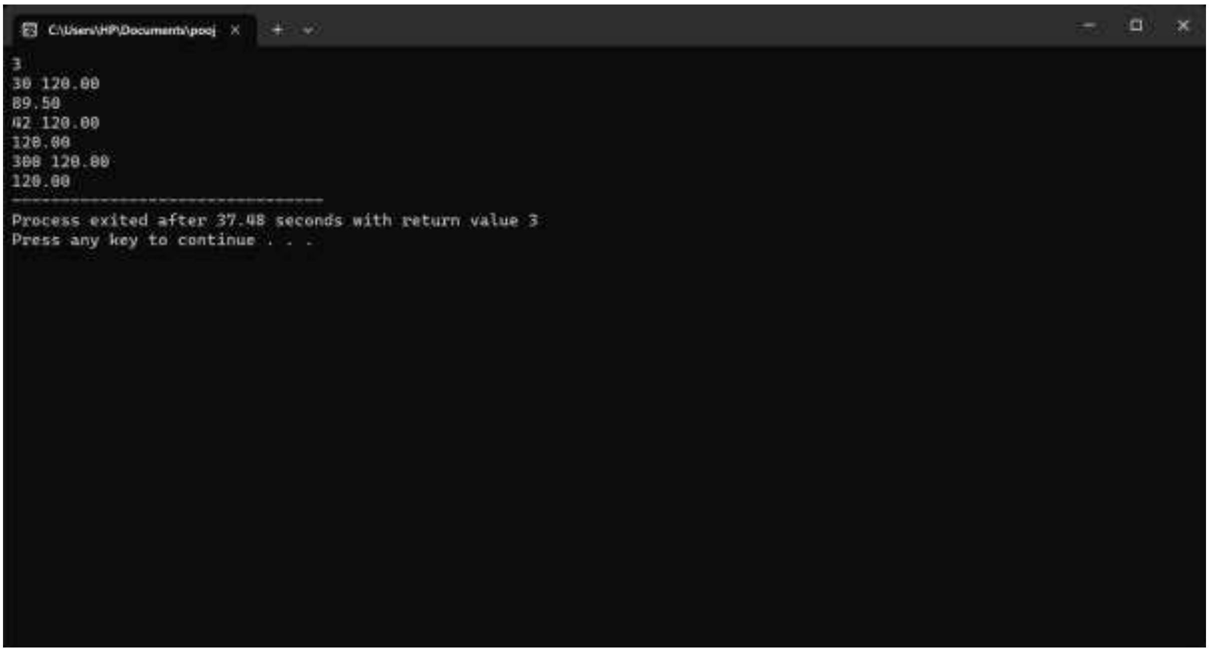
PSEUDOCODE:

```
BEGIN
DECLARE x,i,t,*px
DECLARE FLOAT y,*py,c
GET t
FOR(int i=0;i<t;i++)
    GET x and y
    Px=&x and py=&y
    IF(*px%5==0)
        C=*py-(float)*px-0.5
            IF(c<0)
                PRINT *py
            ELSE
                PRINT c
    ELSE
        PRINT *py
END FOR
END
```

SOURCE CODE :

```
#include <stdio.h>
int main()
{
    int x,i,t,*px; float y,*py,c;
    scanf("%d",&t);
    for(i=0;i<t;i++)
    {
        scanf("%d%f",&x,&y);
         px=&x;
        py=&y;
        if(*px%5==0)
        {
            c=*py-(float)*px-0.5;
            if(c<0)
                printf("%.2f",*py);
            else
            printf("%.2f",c);
        }
        Else
        {
            printf("%.2f",*py);
        }
    }
}
```

OUTPUT:



```
3
30 120.00
89.50
42 120.00
120.00
300 120.00
120.00
--------------------------------
Process exited after 37.48 seconds with return value 3
Press any key to continue . . .
```

RESULT:

Thus the program for finding pooja’s account balance using pointers is executed successfully and the output is verified.

Ex.No : 1.1 (c)

Date :

TO CHANGE THE VALUE OF CONSTANT INTEGER

AIM:

To write a C-program to change the value of constant integer.

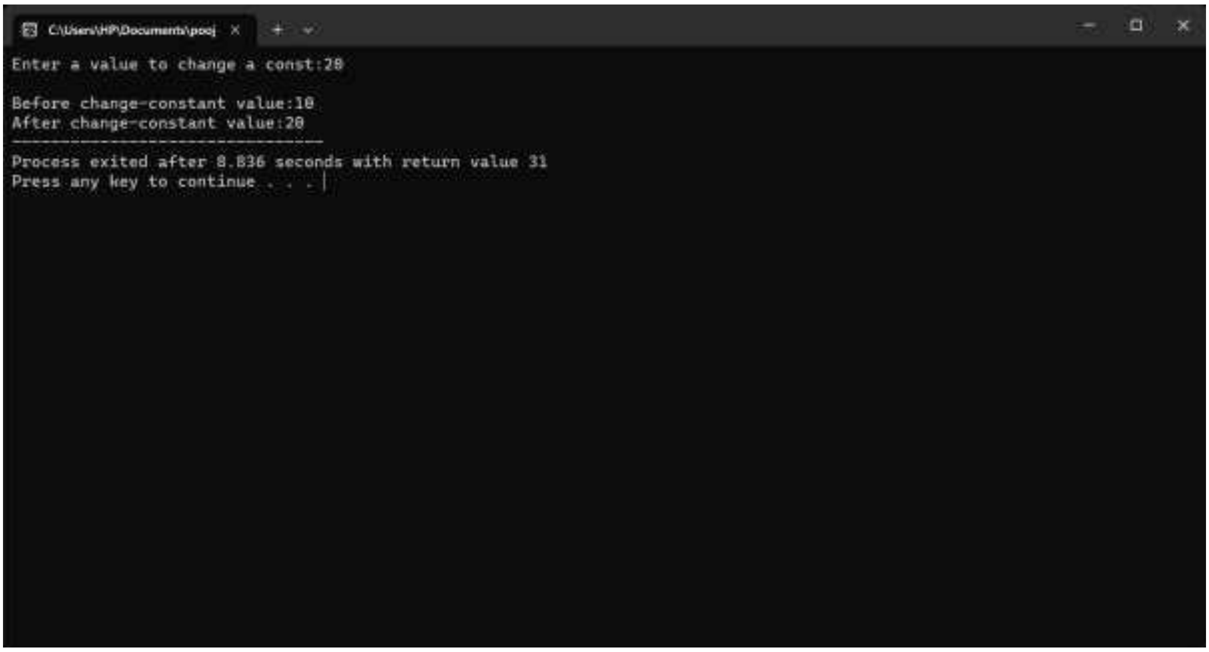
PSEUDOCODE:

```
BEGIN
DECLARE constant a=10 as a integer
DECLARE *ap,b as a integer
Print enter a value to change constant
Get b
Print Before change-constant value a
Initialize ap=&a,*ap=b
Print After change-constant value a
END
```

SOURCE CODE :

```
#include <stdio.h>
int main()
{
const int a=10;
int *ap,b;
printf("Enter a value to change a const:");
scanf("%d",&b);
printf("\nBefore change-constant value:%d",a);
ap=&a;
*ap=b;
printf("\nAfter change-constant value:%d",a);
}
```

OUTPUT:



```
Enter a value to change a const:20

Before change-constant value:10
After change-constant value:20
--------------------------------
Process exited after 8.836 seconds with return value 31
Press any key to continue . . .
```

RESULT:

Thus the program for changing the value of constant using pointer is executed successfully and the output is verified.

Ex.No : 1.1 (d) Date :	**TO CHECK COURSE IS REGISTERED OR NOT**

AIM:

To determine the friends would register the course or not.

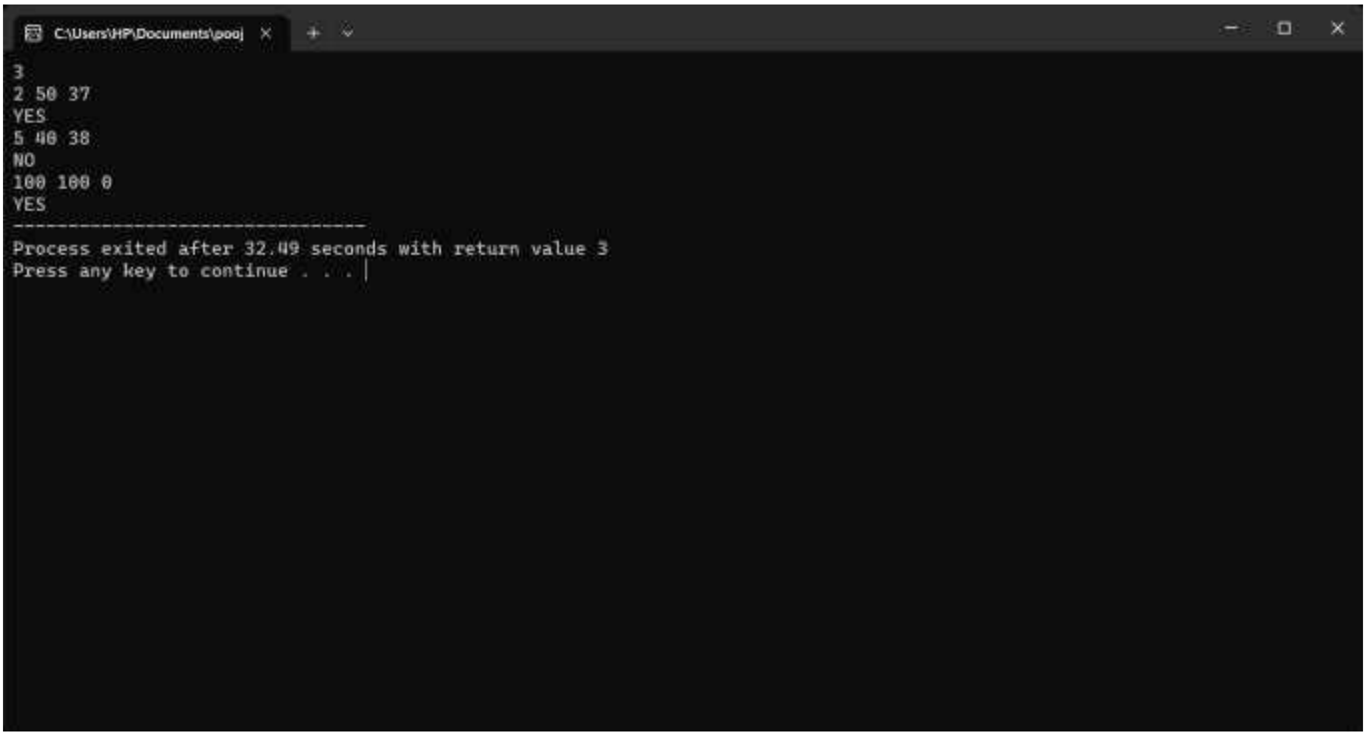
PSEUDOCODE:

```
BEGIN
DECLARE i=0,t,n,m,k,*pn,*pm,*pk,a as integer
Get t
FOR(i=0;i<t;i++)
        GET n,m,k
        ASSIGN address of n,m,k to *pn,*pm,*pk a=n+k
        If a<=m
                Print YES
        Else
        Print NO
        END IF
END FOR
END
```

SOURCE CODE :

```
#include <stdio.h>
 int main()
{
        int i,t,n,m,k,a;
        int *pn,*pm,*pk;
        scanf("%d",&t);
        for(i=0;i<t;i++)
        {
                scanf("%d%d%d",&n,&m,&k);
                 pn=&n;
                pm=&m;
                pk=&k;
                a=n+k;
                if(a<=m)
                        printf("YES");
                else printf("NO");
        }
    }
```

OUTPUT:



```
3
2 50 37
YES
5 40 38
NO
100 100 0
YES
--------------------------------
Process exited after 32.49 seconds with return value 3
Press any key to continue . . .
```

RESULT:

Thus the program to determine it will be possible for all the N friends to register for thecourse or not is executed successfully and the output is verified.

Ex.No : 1.2 (a)

Date :

PEAK ELEMENT OF THE ARRAY

AIM:

Find a peak element i.e. an element that is not smaller than its neighbours.

PSEUDOCODE:

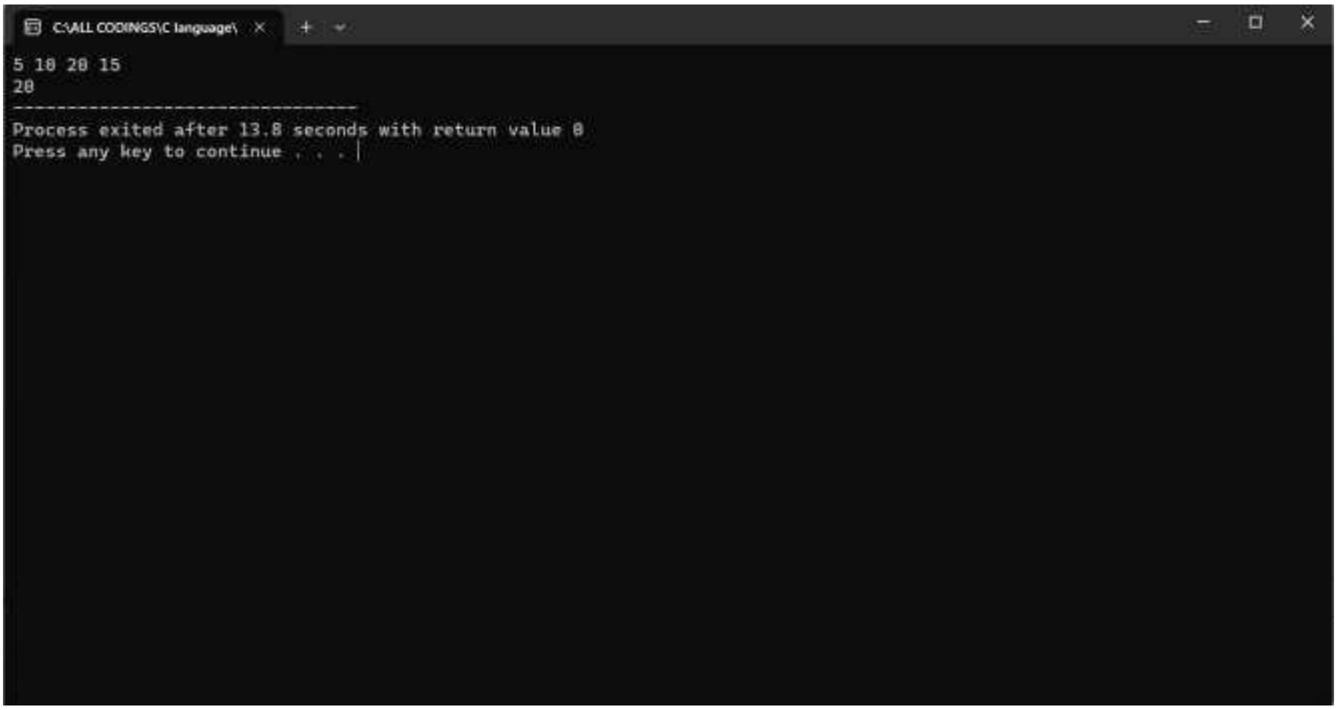
```
BEGIN
DECLARE a[1000] , i=0 , ch=’ ‘

WHILE(ch==’ ‘)
            GET n
   ch a[i++]=n
WHILE END
FOR(int i=0;i<i-1;i++)
             IF(a[j]>a[j-1] && a[j]>a[j+1])
            PRINT a[j]
    END IF
END FOR

END
```

SOURCE CODE :

```
 #include<stdio.h>
int main()
{
        int a[1000],i=0,n;
        char ch=' ';

        while(ch==' ')
        {
                scanf("%d%c",&n,&ch);
                a[i++]=n;
        }
```

```
   for(int j=1;j<i-1;j++)
       {
            if(a[j]>a[j-1] && a[j]>a[j+1])
            {
                        printf("%d ",a[j]);
                        }
            }
}
```

OUTPUT:



```
5 10 20 15
20
--------------------------------
Process exited after 13.8 seconds with return value 0
Press any key to continue . . .
```

RESULT:

Thus the program successfully executed of peak element of the array and the output is verified.

Ex.No : 1.2 (b) Date :	SUM OF EVERY ELEMENTS EXCEPT AT THAT INDEX

AIM:

To find the sum of every array number , except the number at that index.

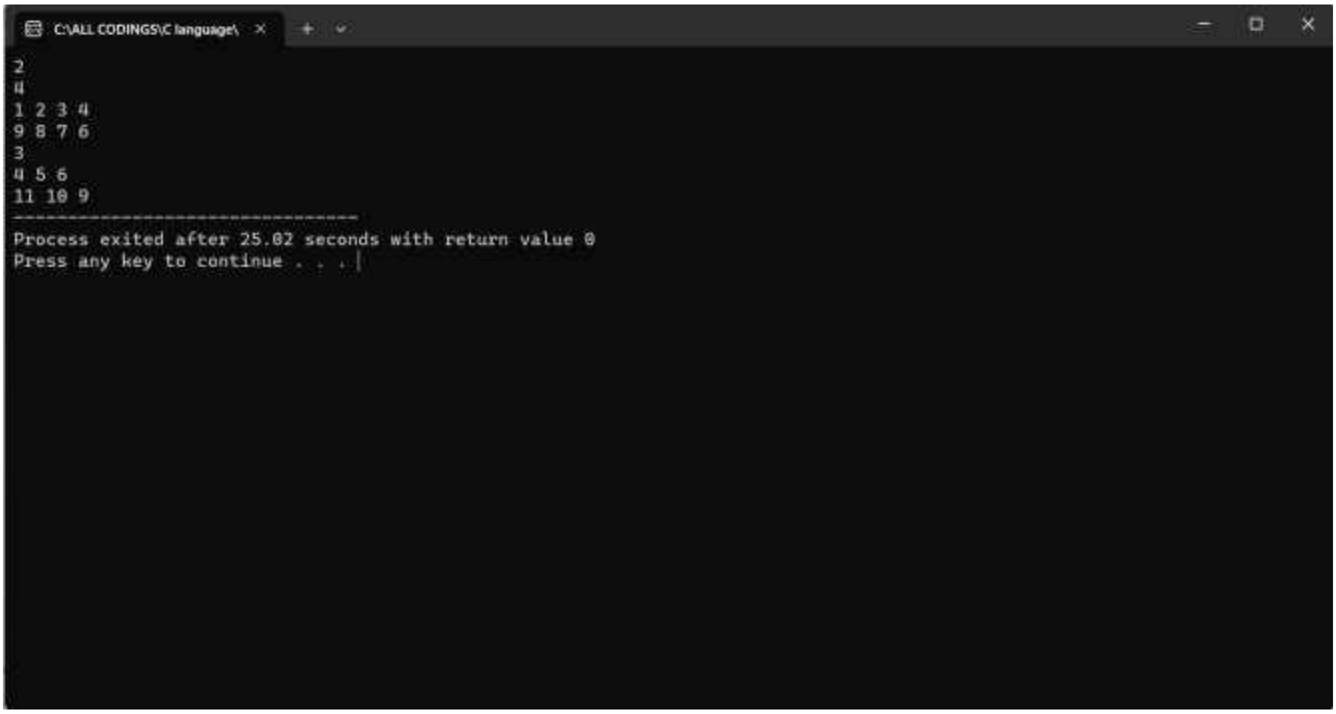
PSEUDOCODE:

```
BEGIN
DECLARE p
GET p
WHILE(p!=0)
    DECLARE n
    GET n
    DECLARE a[n],sum=0
    FOR(int i=0;i<n;i++)
        GET a[i]
        Sum=sum+a[i]
    END FOR
    FOR(int j=0;j<n;j++)
        PRINT sum-a[j]
    END FOR
    p--
END WHILE
END
```

SOURCE CODE :

```
#include<stdio.h>
int main()
{
    int p;
    scanf("%d",&p);

    while(p!=0)
    {
        int n;
        scanf("%d",&n);
        int a[n];
        int sum=0;

        for(int i=0;i<n;i++)
        {
            scanf("%d",&a[i]);
            sum=sum+a[i];
        }

        for(int j=0;j<n;j++)
        {
            printf("%d ",sum-a[j]);
        }
        p--;
    }
}
```

OUTPUT:



```
2
4
1 2 3 4
9 8 7 6
3
4 5 6
11 10 9
--------------------------------
Process exited after 25.02 seconds with return value 0
Press any key to continue . . .
```

RESULT:

Thus the program displayed the sum of every element except the index of that element in array and the output is verified.

Ex.No : 1.2 (c) Date :	DISPLAY THE INDEXES OF SUB-ARRAY

AIM:

To Find a continuous sub-array that adds to a given number S and return the left and right index(1-based indexing) of that subarray.

PSEUDOCODE:

```
BEGIN
DECLARE n
GET n
DECLARE a[n] , s
GET S
FOR(int i=0;i<n;i++)
    GET a[i]
END FOR
FOR(int i=0;i<n;i++)
    DECLARE sum=a[i],c=0;
    FOR(int j=i+1;j<n;j++)
        Sum=sum+a[j]
        IF(sum==s)
            PRINT i+1 and j+1
            C++;
        END IF
    END FOR
    Sum=0
    IF(c==1)
    BREAK
```

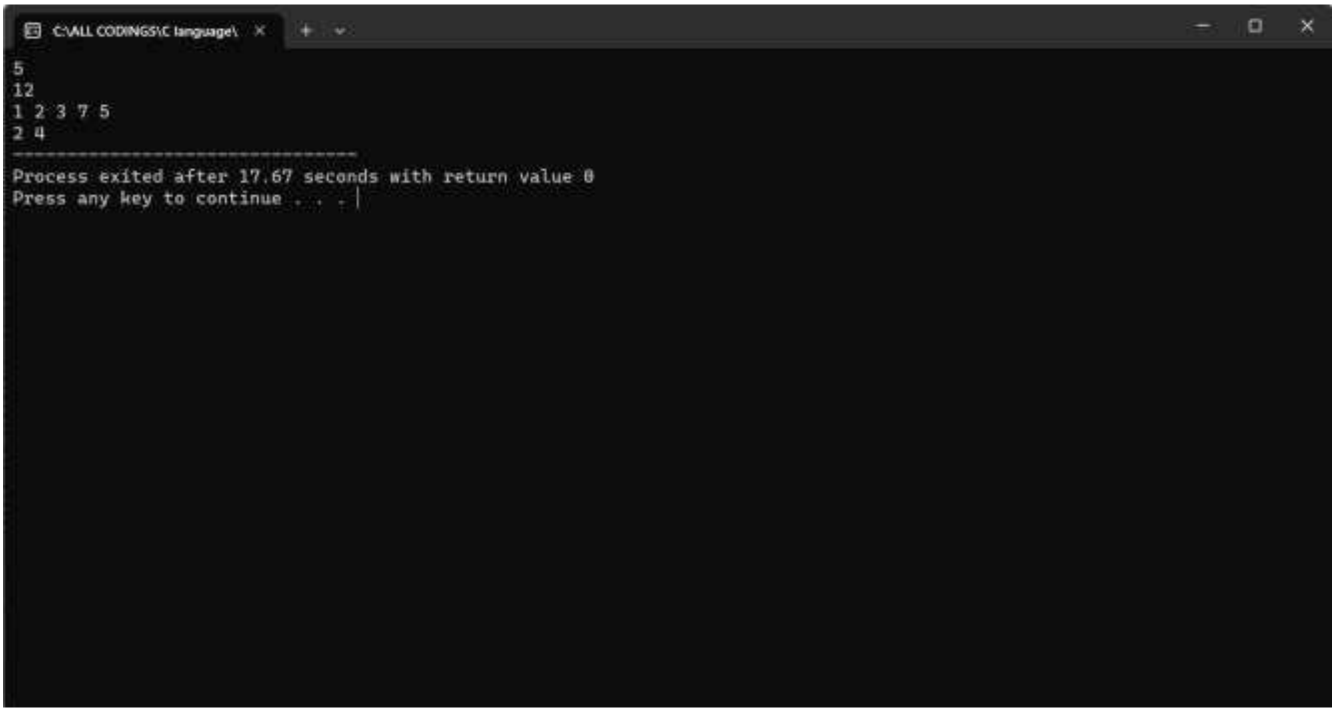
717822F124

```
END FOR
END
```

SOURCE CODE :

```
#include<stdio.h>
int main()
{
        int n;
        scanf("%d",&n);
        int a[n],s;
        scanf("%d",&s);

        for(int i=0;i<n;i++)
        {
                scanf("%d",&a[i]);
        }

        for(int i=0;i<n;i++)
        {
                int sum=a[i],c=0;

                for(int j=i+1;j<n;j++)
                {
                        sum=sum+a[j];
                        if(sum==s)
                        {
                        printf("%d %d",i+1,j+1);
                        c++;
                   }
                }
                sum=0;
                if(c==1)
                break;
        }
}
```

OUTPUT:



```
5
12
1 2 3 7 5
2 4
--------------------------------
Process exited after 17.67 seconds with return value 0
Press any key to continue . . .
```

RESULT:

Thus the program executed a continuous sub-array that adds to a given number S successfully and the output is verified.

Ex.No : 1.2 (d) Date :	DISPLAY THE REPEATED ARRAY ELEMENTS

AIM:

To find all the elements occurring more than once in the given array.

PSEUDOCODE:

```
BEGIN
DECLARE n,c=0
GET n
DECLARE a[n]
FOR(i=0;i<n;i++)
    GET a[i]
END FOR
FOR(int i=0;i<n;i++)
    FOR(int j=i+1;j<n;j++)
        DECLARE t=a[i]
        a[i]=a[j]
        a[j]=t
    END FOR
END FOR

FOR(int i=0;i<n;i++)
    FOR(int j=i+1;j<n;j++)
        IF(a[i]==a[j])
            PRINT a[i]
            C++;
        END IF
    END FOR
```

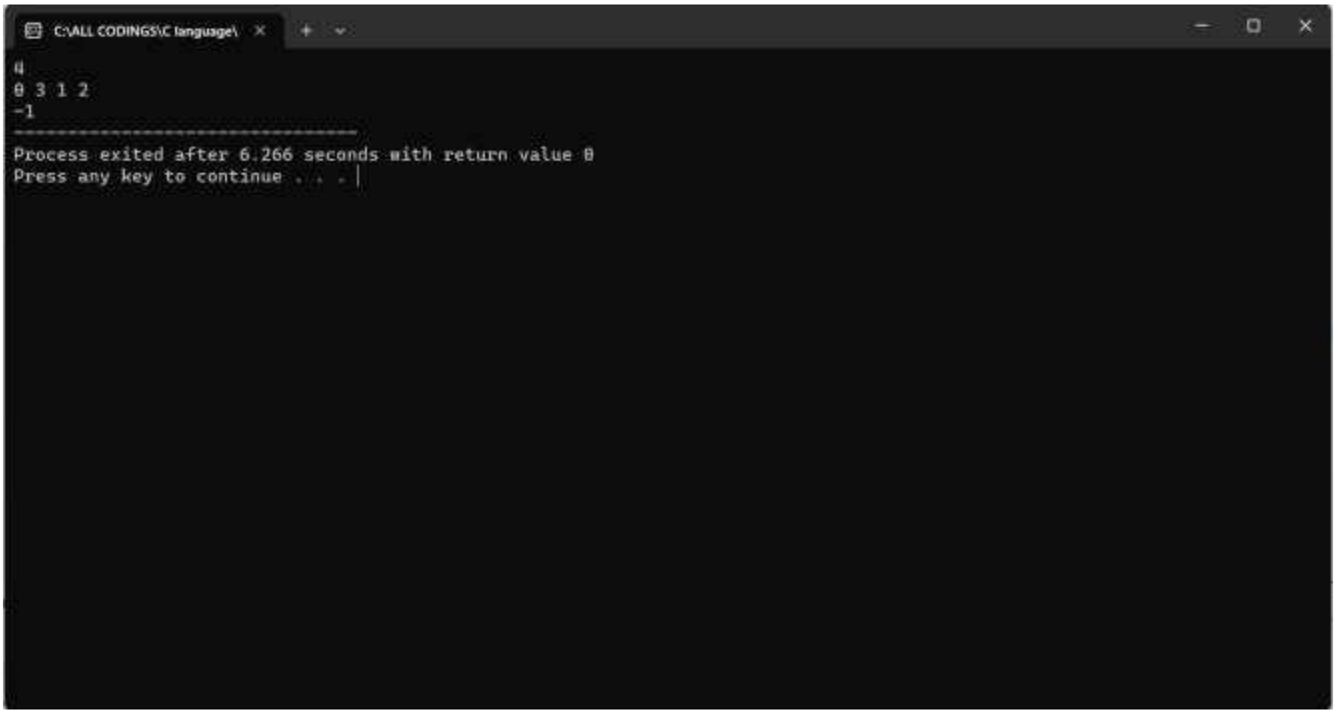
717822F124

```
        END FOR
        IF(c==0)
        PRINT “-1”
        END
```

SOURCE CODE :

```
#include<stdio.h>
int main()
{
        int n,c=0;
        scanf("%d",&n);
        int a[n];

        for(int i=0;i<n;i++)
        {
            scanf("%d",&a[i]);
        }

        for(int i=0;i<n;i++)
        {
            for(int j=i+1;j<n;j++)
            {
                int t=a[i];
                a[i]=a[j];
                a[j]=t;
            }
        }

        for(int i=0;i<n;i++)
        {
            for(int j=i+1;j<n;j++)
            {
                  if(a[i]==a[j])
                  {
                        printf("%d ",a[i]);
                        c++;
            }
            }
        }

        if(c==0)
        printf("-1");
}
```

OUTPUT:



```
4
0 3 1 2
-1
--------------------------------
Process exited after 6.266 seconds with return value 0
Press any key to continue . . . 
```

RESULT:

Thus the program executed to finding all the elements occurring more than once in the given array successfully and output is verified.

717822F124

Ex.No : 2.1 (a) Date :	**FIND THE HIDDEN WORD OF STIRING**

AIM:

To determine the string M, Given the hidden word S and guess T.

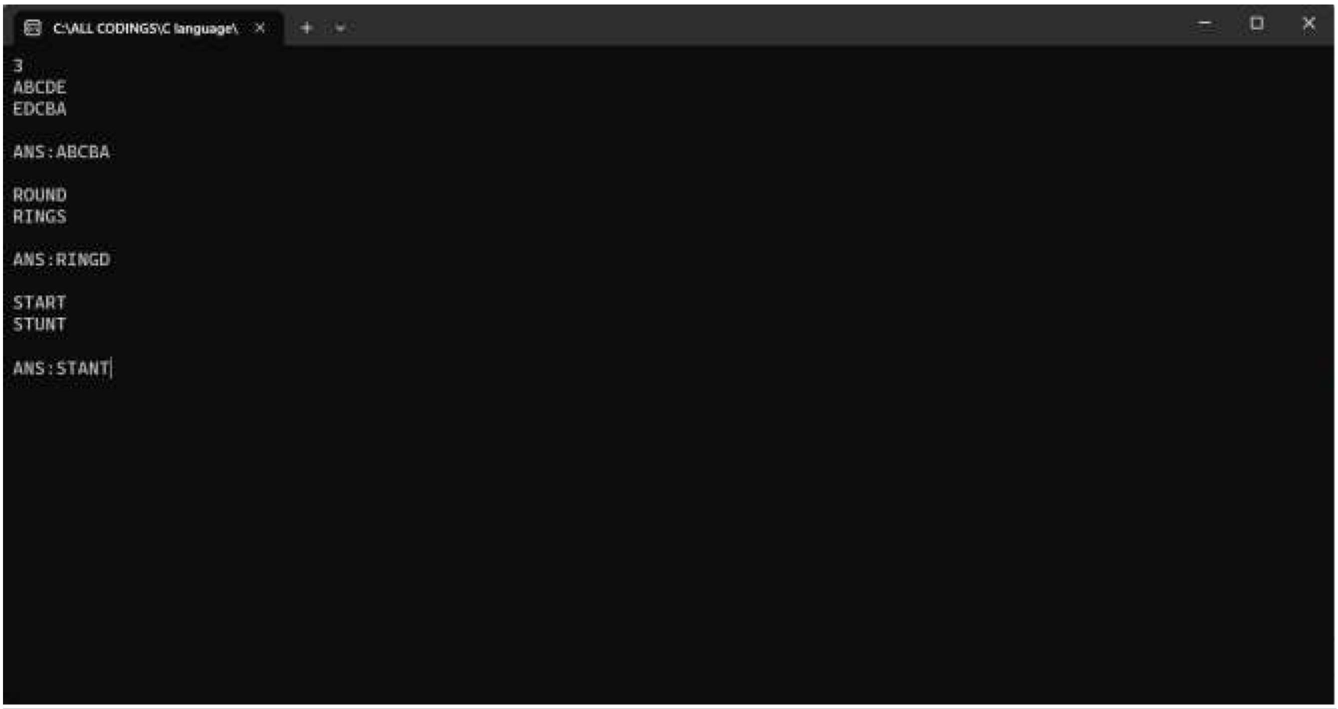
PSEUDOCODE:

```
BEGIN
DECLARE p
GET p

WHILE(p!=0)
    DECLARE  n=5
    DECLARE s[n],t[n],*ps,*pt
    GET s,t
    ASSIGN ps=s and pt=t
    FOR(int i=0;i<n;i++)
        IF(*ps==*pt)
            PRINT *ps
        ELSE
            *ps>*pt ? PRINT *pt : PRINT *ps
        END IF
        Ps++
        Pt++
    END FOR
EDND WHILE
END
```

SOURCE CODE :

```
#include<stdio.h>
int main()
{
   int p;
   scanf("%d",&p);

   while(p!=0)
   {
              int n=5;
              char s[n],t[n],*ps,*pt;

              scanf("%s%s",s,t);
              ps=s;
              pt=t;

              for(int i=0;i<n;i++)
              {
                         if(*ps==*pt)
                         {
                                    printf("%c",*ps);
                         }
                         else if(*ps!=*pt)
                         {
                                    *ps>*pt?printf("%c",*pt):printf("%c",*ps);
                         }

                         ps++;
                         pt++;
              }
   }
}
```

OUTPUT:



```
3
ABCDE
EDCBA

ANS:ABCBA

ROUND
RINGS

ANS:RINGD

START
STUNT

ANS:STANT
```

RESULT:

Thus the program successfully executed of finding the hidden string and the output is verified.

717822F124

Ex.No : 2.1 (b) Date :	**TO FIND THE LARGEST SUBSEQUENCE OF STRING**

AIM:

To print the size of largest subsequence of the string A such that all the characters in the subsequence are distinct.

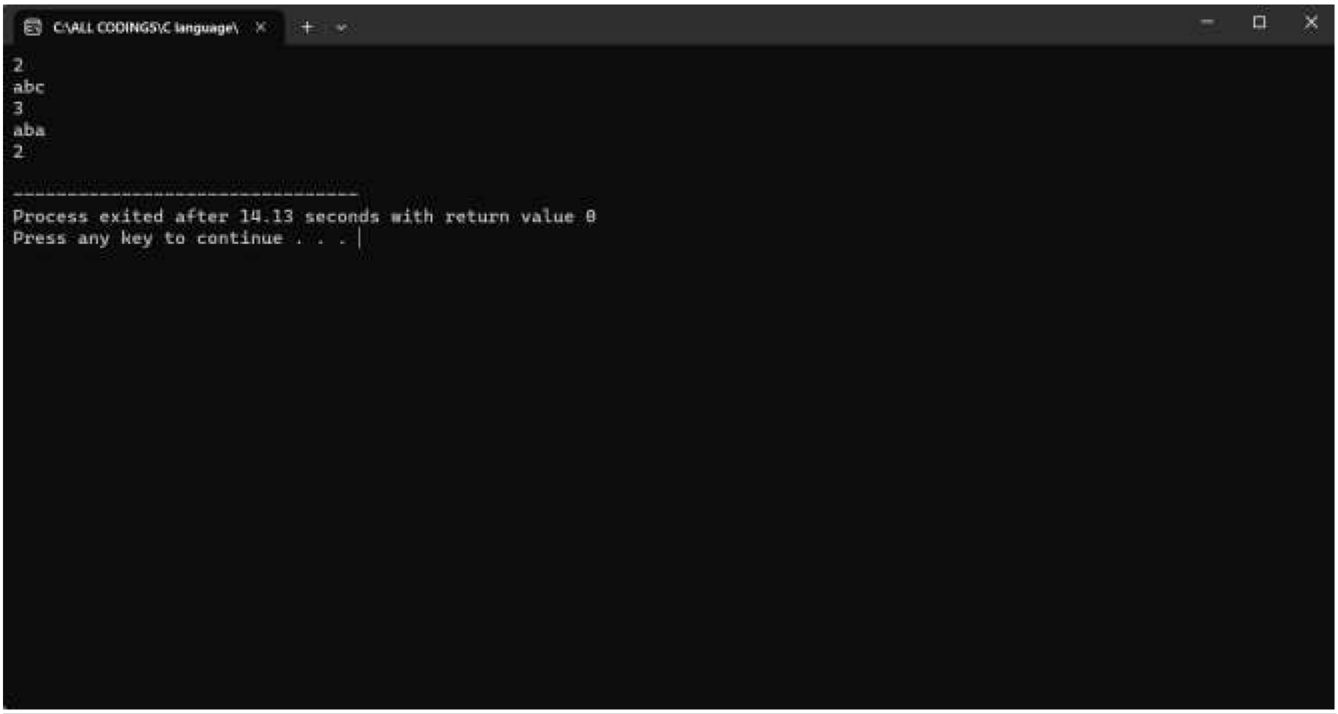
PSEUDOCODE:

```
BEGIN
DECLARE p
GET  p
WHILE (p!=0)
    DECLARE ch[100],*pch
    GET ch
    Pch=ch
    DECLARE c=1 , l=strlen(ch) , t=0
    FOR(int oi=0;i<l;i++)
        IF(*pch+1==*(pch+1))
            C++
        ELSE
            IF(c>t)
                t=c
                c=0
        END IF
        Pch++
    END FOR
    PRINT t
    P - -
END WHILE
END
```

SOURCE CODE :

```
#include<stdio.h>
#include<string.h>
int main()
{
  int p;
  scanf("%d",&p);

  while(p!=0)
  {
     char ch[100],*pch;
     scanf("%s",ch);
     pch=ch;
     int c=1,l=strlen(ch),t=0;

     for(int i=0;i<l;i++)
     {
            if(*pch+1==*(pch+1))
            {
                      c++;
                      }
                      else
                      {
                                if(c>t)
                                t=c;
                                c=0;
                      }
                      pch++;
            }


            printf("%d\n",t);
            p--;
  }
}
```

OUTPUT:



```
2
abc
3
aba
2

--------------------------------
Process exited after 14.13 seconds with return value 0
Press any key to continue . . .
```

RESULT:

Thus the program successfully executed the print the size of largest subsequence of the string A and the output is verified.

Ex.No : 2.1 (c) Date :	**PRINT THE NUMBER OF SPECIAL INGREDIENTS**

AIM:

To print the number of special ingredients in the dish in a given constrained cases.

PSEUDOCODE:

```
DECLARE t
GET t
WHILE(t--)
    DECLARE n, i , j, x=0
    GET n
    DECLARE s[n][500]
    FOR(int i=0;i<n;i++)
        GET s[i]
    END FOR
    FOR(int i=’a’;i<=’z’;i++)
        FOR(int j=0;j<n;j++)
            DECLARE *p
            P=strchr(s[j],i)
            IF(p==NULL)
                BREAK
            ENDIF
        ENDFOR
        IF(j==n)
            X++
        ENDIF
    ENDFOR
```

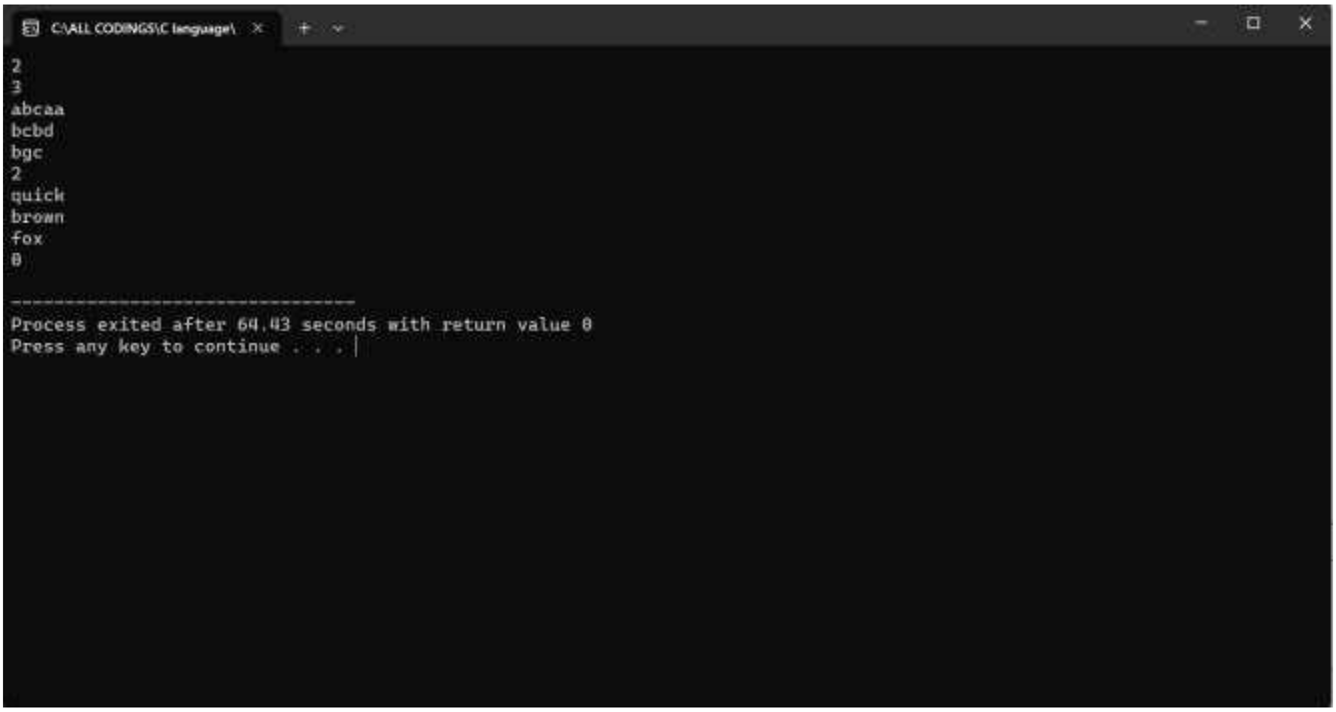
```
        PRINT x
    END WHILE
    END
```

SOURCE CODE :

```
#include  <stdio.h>
#include <stdlib.h>
#include <string.h>
int main(void)
{
   int t;
   scanf("%d",&t);

   while(t--)
         {
                  int n, I, j, x=0;
                  scanf ("%d",&n);
                  char s[n][500];
                  for(I=0;I<n;I++)
                  {
                     scanf ("%s",s[I]);
                  }
                  for(I='a';I<='z';I++)
                  {
                  for(j=0;j<n;j++)
                  {
                     char *p;
                     p=strchr(s[j],I);
                     if(p==NULL)
                     break;
```

```
                }
                if(j==n)
                x++;
                }


    printf ("%d\n",x);
    }
return 0;
}
```

OUTPUT:



```
2
3
abcaa
bcbd
bgc
2
quick
brown
fox
0

--------------------------------
Process exited after 64.43 seconds with return value 0
Press any key to continue . . .
```

RESULT:

Thus the program successfully executed the print the number of special ingredients in a dish and the output is verified.