Ex. No.:

Date:

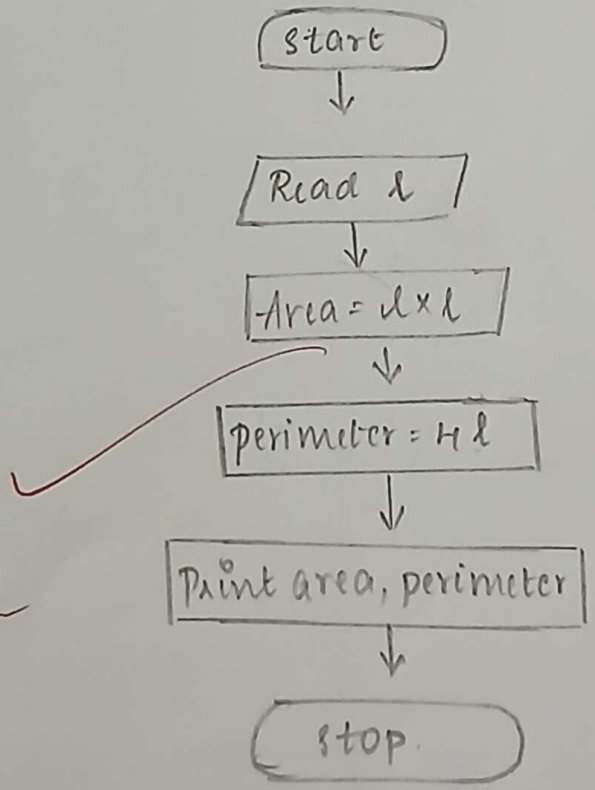
Calculate Area and Perimeter

Write an Algorithm and draw a Flowchart to Calculate the area and perimeter of a square.

Algorithm:

STEP-1: Start
STEP-2: Read l
STEP-3: Compute l x l
STEP-4: Perimeter = 4l
STEP-5: Display area, perimeter.
STEP-6: Stop

Flowchart:



Ex. No.: **Date:**

Days to Year Conversion

Write an Algorithm and draw a Flowchart to convert the given days into years & months.

Algorithm:

STEP-1: Start

STEP-2: Input no. of days

STEP-3: Calculate no of years, years = days / 365

STEP-4: Calculate remaining days, remaining days = days % 365

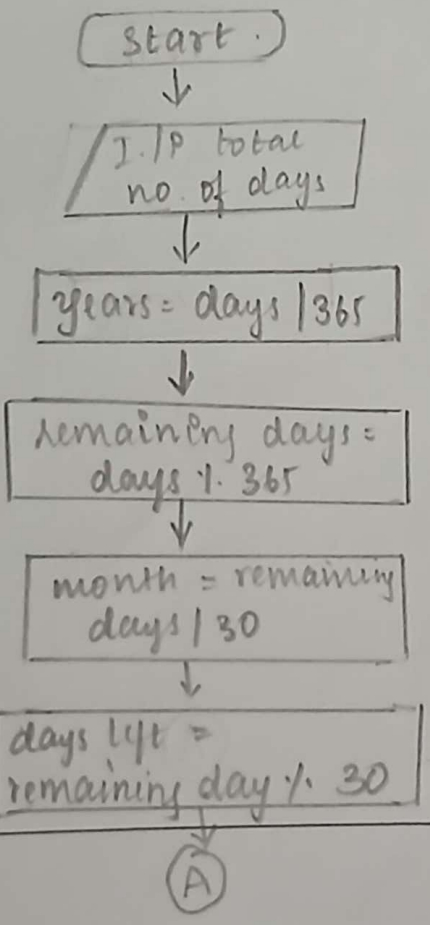
STEP-5: Find no. of moths, months = remaining days / 30

STEP-6: Find remaining days after cacualting months, days left = remaining days % 30

STEP-7: Display years, months of days left.

STEP-8: Stop.

Flowchart:



Ex. No.:

Date:

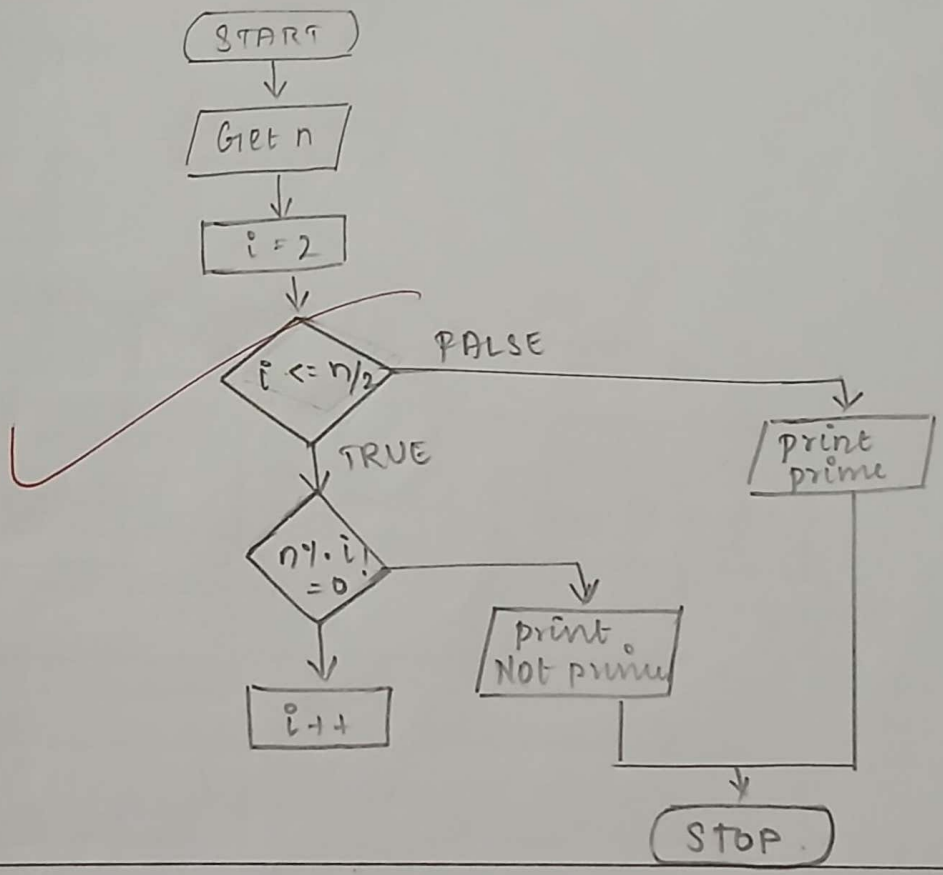
Prime Number

Write an Algorithm and draw a Flowchart to check whether the given number is Prime or not.

Algorithm:

STEP-1: START
STEP-2: Get n
STEP-3: Set i = 2
STEP-4: Check if i<=n/2 then go to step 6 else go to step 5
STEP-5: Print "Not prime" goto step 8.
STEP-6: If n%i==0, print "Not prime" else print "Prime".
STEP-7: Repeat step 6 & 7 until i<=sqrt(n)
STEP-8: STOP

Flowchart:



Ex. No.:

Date:

Leap Year

Write an Algorithm and draw a Flowchart to check whether the given year is Leap year or not.

Algorithm:

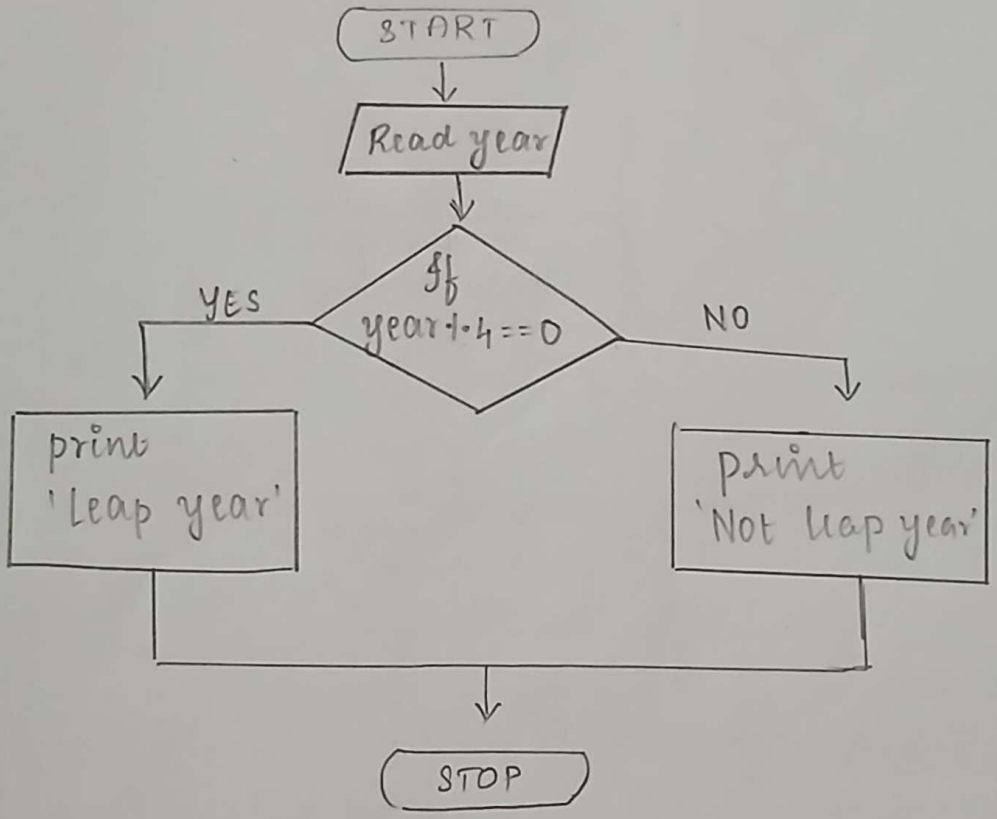
STEP-1: START

STEP-2: Read year

STEP-3: If (y%4==0) then print 'leap year' else print 'Not leap year'.

STEP-4: STOP

Flowchart:



Ex. No.: **Date:**

Palindrome Number

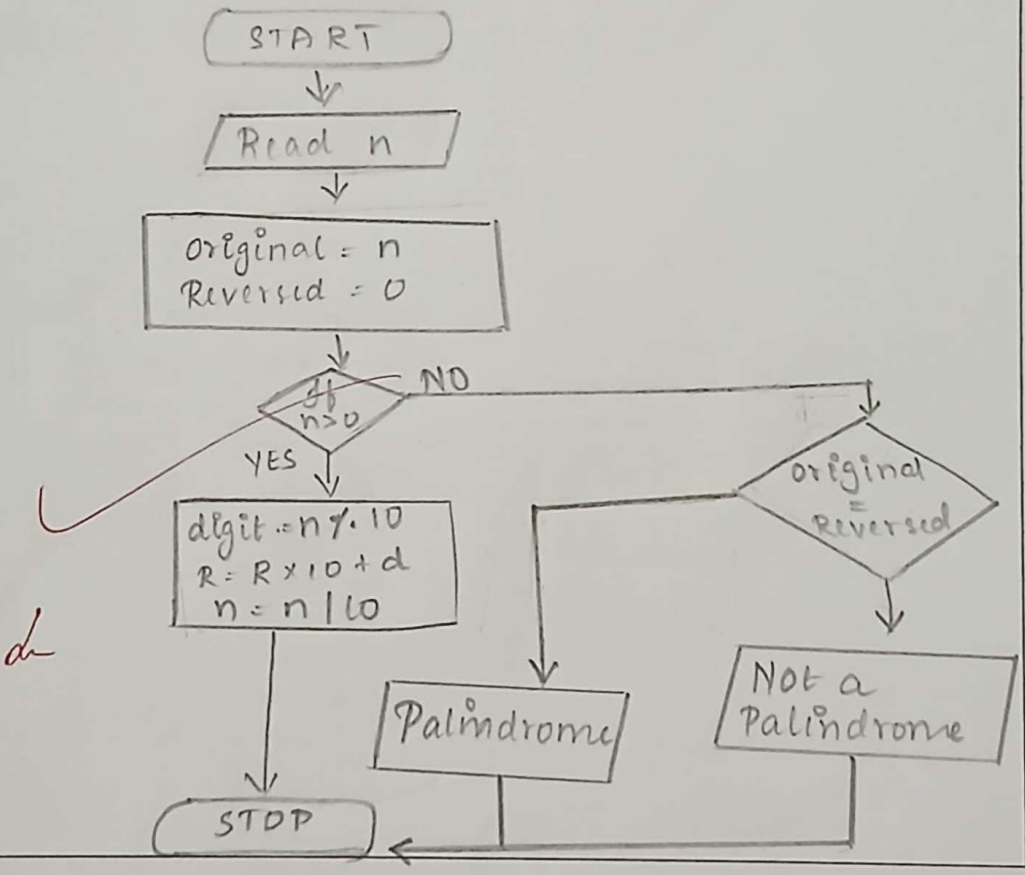
Write an Algorithm and draw a Flowchart to check whether the given number is palindrome number or not.

Algorithm:

STEP-1: Start
STEP-2: Read the number, n
STEP-3: Initialise, set original = n & reversed = 0
STEP-4: while n>0
- set digit = n mod 10
- update reversed = reversed x 10 + digit
- update n = n ÷ 10

STEP-5: If original = reversed, print 'palindrome'.
STEP-6: Else print 'Not a palindrome'.
STEP-7: STOP.

Flowchart:



Ex. No.: **Date:**

Sum of Digits

Write an Algorithm and draw a Flowchart to calculate the sum of digits in the given number.

Algorithm:

STEP-1: Start

STEP-2: Input the number

STEP-3: Initialise sum = 0

STEP-4: Repeat the following steps while n is greater than 0. n>0. Extract the last digit of n: digit = n%10. Add the digits to sum

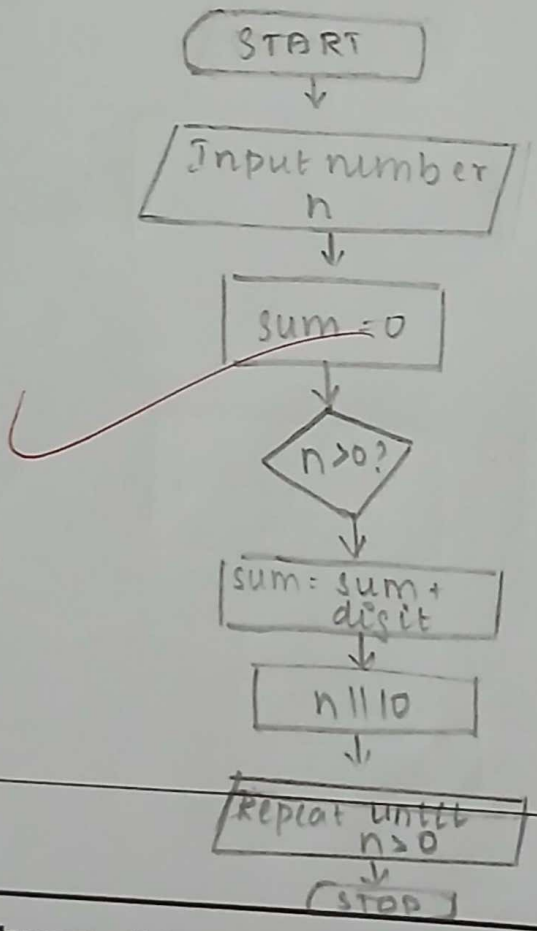
sum = sum + digit

Remove the last digit & form n.

STEP-5: print sum

STEP-6: STOP

Flowchart:



Overview of C, Constants, Variables and Data Types

Ex. No.: **Date:**

Say "Hello, World!" With C

Problem Statement:

This is a simple challenge to help you practice printing to stdout.
We're starting out by printing the most famous computing phrase of all time! In the editor below, use either printf or cout to print the string Hello, World! to stdout.

Input Format
You do not need to read any input in this challenge.

Output Format
Print ***Hello, World!*** to stdout.

Sample Output 1
Hello, World!

Program:

```
#include<stdio.h>
int main() {
    printf("Hello, World!");
    return 0;
}
```

1

2.

Program:

```
#include<stdio.h>
int main()
{ char ch;
  scanf("%c", &ch);
  printf("%c", ch);
  return 0;
}
```

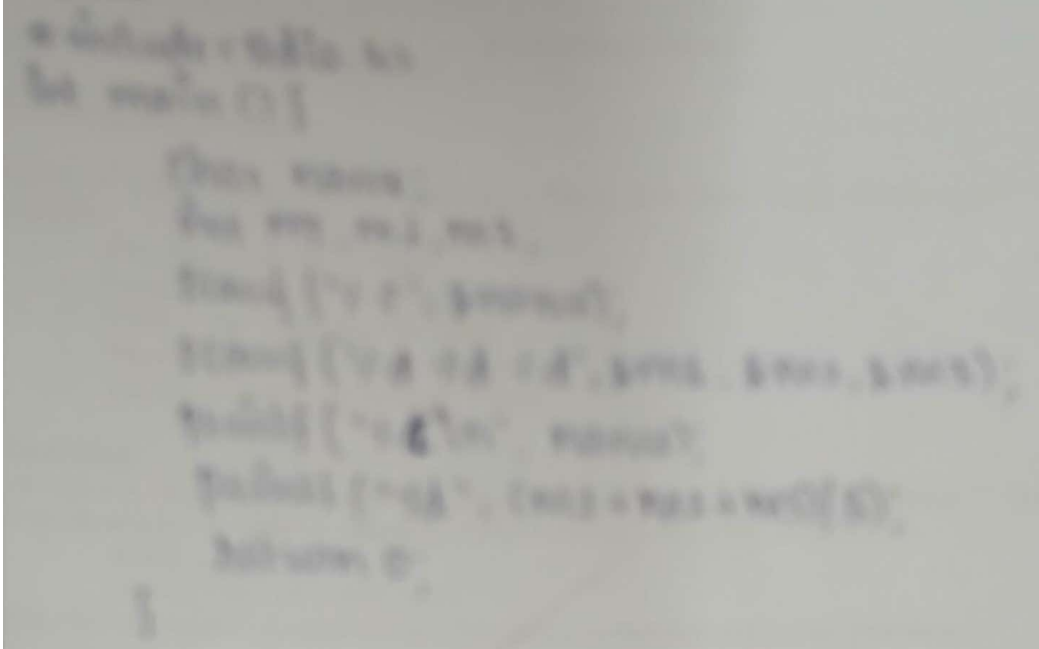
Program:

```
#include<stdio.h>
int main() {
    int a,b;
    float c,d;
    scanf("%d %d",&a,&b);
    printf("%d %d\n",
    scanf("%d %d",&c,&d);
    printf("%d %d\n" a+b, a-b);
    printf("%.1f %.1f", c+d, c-d);
    return 0;
}
```

2

2

Program:



ogram:

```
include <stdio.h>
nt main() {
    int a;
    long b;
    char c;
    float d;
    double e;
    scanf("%d %ld %c %f %lf", &a, &b, &c, &d, &e);
    printf("%d\n", a);
    printf("%ld\n", b);
    printf("%c\n", c);
    printf("%.3f\n", d);
    printf("%.9lf\n", e);
    return 0;
}
```

Program:

```
#include <stdio.h>
int main(){
    char ch;
    scanf("%c",&ch);
    printf("%d\n%c %c",ch,ch-1,ch+1);
    return 0;
}
```

2

RESULT: Thus the code is implemented and executed successfully.

Operators and Expressions, Managing Input and Output Operations

Program:

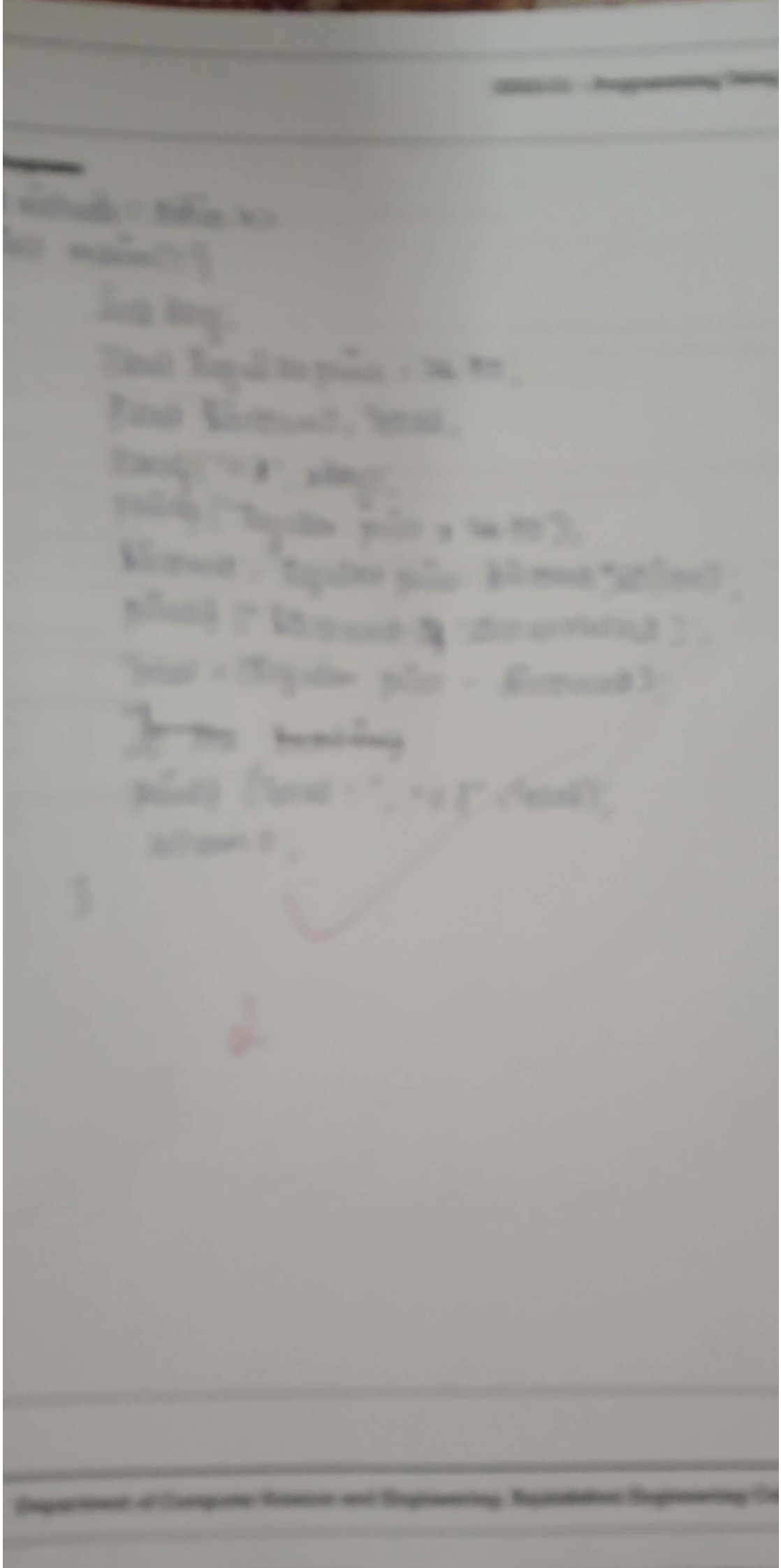
```
#include<stdio.h>
int main(){
        int feet;
        int inch;
        int ht;
        float heightincm;
        scanf("%d",&feet);
        scanf("%d",&inch);
        ht=(12*feet+inch);
        heightincm=(ht*2.54);
        printf("%.2f",heightincm);
        return 0;
}
```

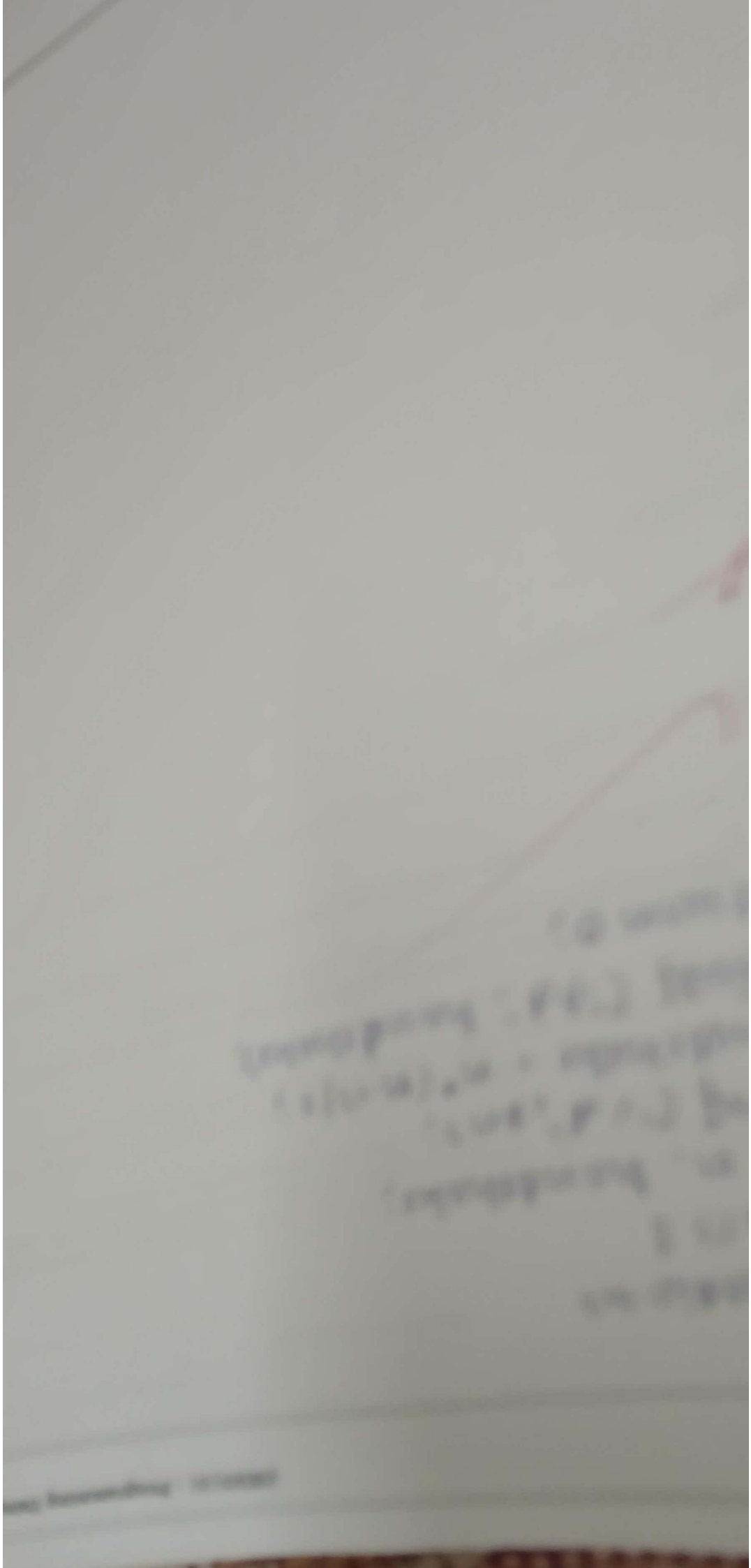
2

2

Program:

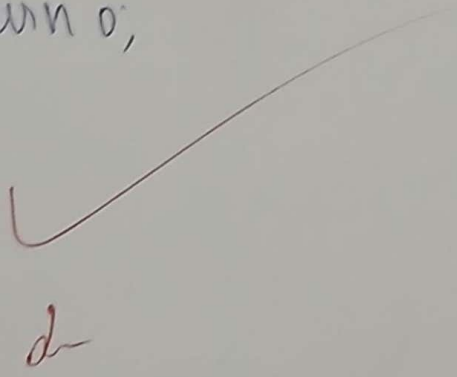
```
#include <stdio.h>
int main(){
        int a,b;
        scanf("%d\n",&a);
        scanf("%d\n",&b);
        printf("%d\n",a+b);
        printf("%d\n",a-b);
        printf("%d\n",a*b);
        printf("%d\n",a/b);
        printf("%d\n",a%b);
        return 0;
}
```





Program:

```
#include <stdio.h>
int main() {
    int a, b, c;
    scanf("%d %d %d", &a, &b, &c);
    if (b>c && b>a){
        printf("%d", b);
    }
    else if (c>a && c>b){
        printf("%d", c);
    }
    else{
        printf("%d", a);
    }
    return 0;
}
```



RESULT: Thus the code is implemented & executed successfully.

Decision Making and Branching – if, if...else, nested if...else, if...else if, Switch-Case

gram:

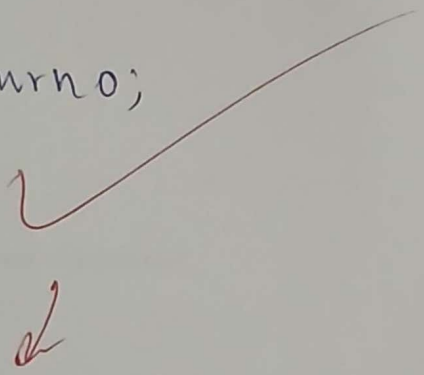
```
#include <stdio.h>
int main (){
    int a, b;
    scanf ("%d %d",&a,&b);
    if (a%10 - b%10){
        printf ("true");
    }
    else {
        printf ("false");
    }
    return 0;
}
```

Program:

```
#include<stdio.h>
int main(){
    int a;
    scanf("%d",&a);
    if(a%2!=0){
        printf("Weird");
    }
    else if(a>2 && a<=5){
        printf("Not Weird");
    }
    else if(a>=6 && a<=20){
        printf("Weird");
    }
    else if(a>20){
        printf("Not Weird");
    }
    return 0;
}
```

ogram:

```
include <stdio.h>
t main() {
    int a, b, c;
    scanf("%d %d %d", &a, &b, &c);
    if (a*a = b*b + c*c) || (b*b == a*a + c*c) || (c*c == a*a + b*b))
    {
      printf("yes");
    }
    else
    {
      printf("no");
    }
    return 0;
}
```



Program:

```
#include <stdio.h>
int main(){
    int a;
    scanf("%d",&a);
    if(a==3){
      printf("Triangle");
    }
    else if(a==4){
      printf("Square");
    }
    else if(a==5){
      printf("Pentagon");
    }
    else if(a==6){
      printf("Hexagon");
    }
    else if(a==7){
      printf("Heptagon");
    }
    else if(a==8){
      printf("Octagon");}
    }
    else if(a==9){
      printf("Nonagon");
    }
```

Program:

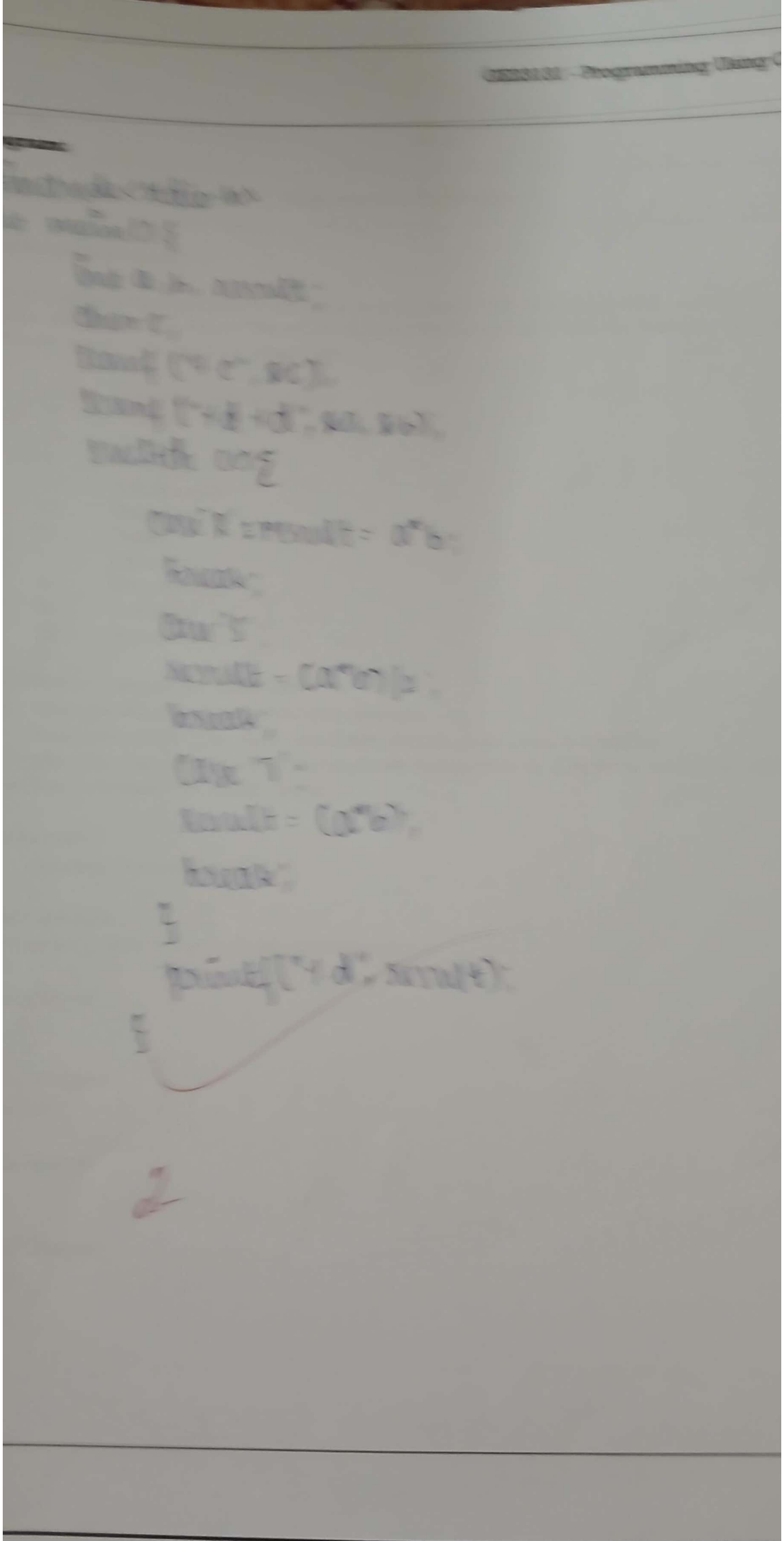
```
#include<stdio.h>
int main(){
    int year;
    scanf("%d",&year);
    int animal = (year-2000) % 12;
    switch (animal) {
    case 0:
    printf("Dragon");
    break;
    case 1:
    printf("Snake");
    break;
    case 2:
    printf("House");
    break;
    case 3:
    printf("Sheep");
    break;
    case 4:
    printf("Monkey");
    break;
    case 5:
    printf("Rooster");
    break;
    case 6:
    printf("Dog");
    break;
    case 7:
    printf("Pig");
```

```
case 8:
printf("Rat");
break;
case 9:
printf("Ox");
break;
case 10:
printf("Tiger");
break;
case 11:
printf("Hare");
break;
}case 12:
}  printf
   break
```

Program:

```
#include<stdio.h>
int main() {
    int d, m, y, feb;
    scanf("%d %d %d", &d, &m, &y);
    if ((y % 100 == 0 && y % 400) || (y % 4 == 0)) {
        feb = 29;
    }
    else {
    feb = 28;
    switch(m) {
        case 1:
        printf("%d", d);
        break;
        case 2:
        printf("%d", 31 + d);
        break;
        case 3:
        printf("%d", 31 + feb + d);
        break;
        case 4;
        printf("%d", 31 + feb + 31 + d);
        break;
        case 5:
        printf("%d", 31 + feb + 31 + 30 + d);
        break;
        case 6:
        printf("%d", 31 + feb + 31 + 30 + 31 + 30 + d);
        break;
```

```
case 7:
printf ("%d", 31+feb+31+30+31+30+d);
break;
case 8:
printf ("%d", 31+feb+31+30+31+30+31+d);
break;
case 9:
printf ("%d", 31+feb+31+30+31+30+31+31+d);
break;
case 10:
printf("%d", 31+feb+31+30+31+30+31+31+30+d
break;
case 11:
printf ("%d", 31+feb+31+30+31+30+31+31+30+31+
break;
case 12:
printf ("%d", 31+feb+31+30+31+30+31+31+30+30+
break;
}
```



Decision Making and Looping – while and do...while, for

Program:

```
#include<stdio.h>
int main(){
    int T, i=0, n, t;
    scanf("%d",&T);
    while(i<T)
    {
      scanf("%d",&n);
      t=n/4
      if(t%2==0 && n%2==0)
      {
        printf("No\n");
      }
      else if(t%2==1 && n%2==1)
      {
        printf("No\n");
      }
      else
      {
        printf("Yes\n");
      }
      i++;
    }
}
```

2

Program:

```
#include <stdio.h>
int main() {
    int a,b,n=0;
    scanf("%d",&a);
    while(a>0)
    {
      b=a%10;
      if (b==0||b==6||b==9||b==4)
      {
        n=n+1;
       }
      else if (b==8)
       {
        n=n+2;
       }
      a=a/10;
    }
    printf("%d",n);
}
```

Program:

```
#include<stdio.h>
int main(){
    int n, r=0;
    scanf("%d", &n);
    while (n!=0)
    {
      n=n/2;
      r=r+1;
    }
    printf("%d", r);
}
```

Program:

```
#include<stdio.h>
int main(){
    int n, r=0;
    scanf("%d", &n);
    while (n!=0)
    {
      n=n/2;
      r=r+1;
    }
    printf("%d", r);
}
```

Program:

```
#include<stdio.h>
int main(){
    int n,x,y=1;
    scanf("%d",&n);
    while(n!=0 && y==1){
        x=n%10; n=n/10;
        if(x==2||x==3||x==4||x==7)
        {
            y++;
        }
    }
    if(y==1){
        printf("true");
    }
    else{
        printf("false");
    }
}
```

Nested Loops - while and for, Jumps in Loops

Program:

```
#include<stdio.h>
int main(){
    int T,d,i=0,i1,i2,o;
    char c;
    scanf("%d",&T);
    while(i<T){
    scanf("%d",&d);
      i1=0;
      while(i1<d){
        o=1
        i2=0
        if(i1%2==0){
          o=0;
        }
        while(i2<d)
        {
          c='B'
          if(i2%2==0){
            c='W';
          }
          printf("%c",c);
          i2++;
        }
        i1+=1;
        printf("\n");
    }
    i=i+1;
}
}
```

Ex. No. :

Date :

Print Our Own Chessboard

Problem Statement:

Let's print a chessboard!

Write a program that takes input:
The first line contains T, the number of test cases
Each test case contains an integer N and also the starting character of the chessboard

Output Format
Print the chessboard as per the given examples

Sample Input:
2
2 W
3 B

Sample Output:
WB
BW
BWB
WBW
BWB

Program:

```
#include<stdio.h>
int main(){
    int T,d,i,i1,i2,o,x;
    char c,s;
    scanf("%d",&T);
    for(i=0;i<T;i++)
    {
     scanf("%d %c",&d,&s);
     for(i1=0;i1<d;i1++);
     {
       x=(s=='W')?0:1;
       o=(i1%2==x)?0:1;
       for(i2=0;i2<d;i2++){
       {
         c=(i2%2==o)?'W':'B';
          printf("%c",c);
       }
       printf("\n");
     }
    }
    return 0;
}
```

2

Ex. No. :

Date :

Armstrong Number

Problem Statement:

The k-digit number N is an Armstrong number if and only if th
sums to N.

Given a positive integer N, return true if and only if it is an Ar

Note: 1 <= N <= 10^8

Hint: 153 is a 3-digit number, and 153 = 1^3 + 5^3 + 3^3.

Sample Input:
153

Sample Output:
true

Sample **Input:**
123

Sample **Output:**
false

Sample **Input:**
1634

Sample **Output:**
true

printf ("true");

printf ("false");

return 0;

```
nt=n; rn=0;
while (n!=0)
{
    rn=rn*10+n%10;
}
n=nt+rn
i++;
}
while (rn!=nt || i==1);
printf ("%d", rn);
return 0;
}
```

2

Program:

```
#include<stdio.h>
int main(){
    int rn, n, nt=0, i=0;
    scanf("%d", &n);
    do {
        nt=n; rn=0;
        while(n!=0)
        {
            rn=rn*10+n%10;
        }
        n=nt+rn
        i++;
    }
    while(rn!=nt || i==1);
    printf("%d", rn);
    return 0;
}
```

2

Program:

```
#include <stdio.h>
int main() {
    int n=1,i=0,nt,co=0,e;
    scanf("%d",&e);
    while (i<e)
    {
      nt=n;
      while(nt!=0)
      {
        co=0;
        if(nt%10!=3 && nt%10!=4)
        {
          co=1;
          break;
        }
      }
      if(co==0)
      {
        i++;
      }
      n++;
    }
    printf("%d",--n);
    return 0;
}
```

RESULT: Thus the code is implemented and executed successfully.

One-Dimensional Arrays

Program:

```
#include <stdio.h>
int main() {
    int t;
    scanf("%d", &t);
    while (t--){
        int n;
        scanf("%d", &n);
        int a[n];
        for (int i=0; i<n; i++)
        {
            scanf("%d", &a[i]);
        }
        int k;
        scanf("%d", &k);
        int flag=0;
        for( int i=0; i<n; i++)
        {
            for (int j=i+1; j<n; j++)
            {
                if (a[i]-a[j]==k || a[j]-a[i]==k)
                {
                    flag=1;
                    break;
                }
            }
            if (flag)
            break;
        }}
        printf("%d\n", flag);
    }
}
```

Program:

```
#include<stdio.h>
int main(){
    int t;
    scanf("%d",&t);
    while(t--){
        int n,c=0;
        scanf("%d",&n);
        for(int i=0;i<=n;i++){
            if(i%2!=0){
                c=c+i;
            }
        }
        printf("%d\n",c);
    }
}
```

2

rogram:

```
#include<stdio.h>
int main(){
    int s1,s2,ans;
    scanf("%d",&s1);
    int ta[s1];
    for(int i=0;i<s1;i++){
    {
        scanf("%d",&ta[i]);
        scanf("%d",&s2);
      {
        int tb[s2];
        for(int i=0;i<s2;i++)
        {
          scanf("%d",&tb[i]);
        }
        for(int j=0;j<s2;j++)
        {
            ans=0;
            for(int i=0;i<s1;i++){
                if(tb[j]==ta[i])
                ans++;
            }
            printf("%d\n",ans);
        }
    }
```

2

Thus the code is executed successfully.

Searching Algorithms – Linear and Binary

Computer Science and Engine

Program:

```
#include <stdio.h>
int main(){
    int t, m, n, c=0;
    scanf("%d", &t);
    for(int i=0; i<t; i++)
    {
        c=0;
        scanf("%d\n %d", &m, &n);
        int arr[n];
        for(int j=0; j<n; j++)
        {
            scanf("%d", &arr[j]);
        }
        for(int a=0; a<n-1; a++)
        {
            for(int b=a+1; b<n; b++)
            {
                if(arr[a]+arr[b]==m)
                {
                    printf("%d %d\n", a+1, b+1);
                    c=1;
                    break;
                }
            }
            if(c==1) break;
        }
    }
    return 0;
}
```

Program:

```
#include <stdio.h>
int main () {
    int n, m, c, c1 = 0, c0;
    scanf ("%d",&n);
    int arr[n];
    for(int a=0; a<n; a++)
    {
        scanf ("%d", &arr[a]);
    }
    scanf ("%d", &m);
    int brr[m], ans[m];
    for (int b=0; b<m; b++) {
        {
            scanf ("%d", &brr[b]);
        }
    for(int j=0 ; j<m; j++)
    {
        c=0;
        for (int i=0; i<n; i++) {
            if (arr[i]==brr[j]) {
                c = 1; arr[i] = -1;
                break;
            }
        }
        if(c==0) {
            ans[c1] = brr[j];
            c1++;
        }
    }
```

Program:

```
#include<stdio.h>
int main(){
    int t, n, Ts, rs, m;
    scanf("%d",&t);
    for(int i=0;i<t;i++){
        Ts=0;
        rs=0;
        scanf("%d",&n);
        int arr[n];
        for(int j=0; j<n; j++)
            scanf("%d",&arr[j]);
        m=n/2;
        if(arr[m]==0){
          for(m=0; arr[m]==0 && m<n; m++);
        }
        for(int j=0; j<=m; j++){
        { Ts=Ts+arr[j];
            for(int j=m; j<n; j++){
              rs=rs+arr[j]
              printf("%s\n",(Ts==rs)?"YES":"NO
            }
        return 0;
    }
```

Sorting Algorithms – Bubble and Selection

Computer Science and Engineering, Rajalakshmi Engineering Colle

ram:

```
#include<stdio.h>
main(){
  int t;
  scanf("%d",&t);
  while(t--)
  {
    int n,m,d,min,temp;
    scanf("%d%d",&n,&m);
    d=n-m;
    int arr[n];
    for(int i=0;i<n;i++)
     scanf("%d",&arr[i]);
    for(int j=0;j<n;j++)
    {
      min=j
      for(int k=j;k<n;k++)
      {
        if(arr[k]<arr[min])
          min=k;
      }
      temp=arr[min]
      arr[min]=arr[j];
      arr[j]=temp;
    }
    int maxsum=0;minsum=0;
    for(int a=0;a<d;a++)
    minsum+arr[a];
    for(int b=n-1;b>m-1;b--)
    maxsum+=arr[b];
    printf("%d\n",maxsum-minsum);
  }
}
```

Program:

```
#include<stdio.h>
int main () {
    int n;
    scanf("%d", &n)
    int arr[n];
    for (int i=0; i<n ; i++)
    scanf ("%d", &arr[i]);
    int max = arr[0];
    for (int i=1 ; i<n; i++)
    {
      if (arr[i] > max)
      max = arr[i];
    }
    max++;
    int min=0;
    for (int a=0 ; a<n; a++)
    {
      for (int b=0; b<n; b++)
      {
        if arr[b] < arr[min])
          min = b;
        }
        printf ("%d", min);
        arr[min] = max;
      }
d-  }
```

Program:

```
#include <stdio.h>
int main(){
    int n, min1, min2, temp, flag=1;
    scanf("%d",&n);
    int vac[n], pat[n];
    for (int i=0; i<n; i++)
     scanf("%d", &vac[i]);
    for(int i=0; i<n; i++)
    {
      min1=j, min2=j;
       for(int k=j; k<n; k++)
       {
         if(vac[k]<vac[min1])
          min1=k;
          if(pat[k]<pat[min2])
          min2=k;
       }
       temp=vac[min];
       vac[min1]=vac[j];
        vac[j]=temp;
       temp=pat[min2];
       pat[min2]=pat[j];
       pat[j]=temp;
  }}
  for(int i=0; i<n; i++)
  {if vac[i]<=pat[i])
   {flag=0;
   break
  }}
```

```
if((flag==1)){
print("yes") }
else{
 printf("No");
 }
}
```

Program:

```
#include<stdio.h>
int main()
{
   int n, count=0;
   scanf("%d",&n);
   int arr[n];
   for(int i=0; i=n-1;i++)
   scanf("%d" &arr[i]);
   for(int i=0; i=n-1;i++)
   {
      if(int j=i+1; j<n; j++)
      {
         if((arr[i]^arr[j])==0)
         count++;
      }
   }
   printf("%d",count);
}
```

```
][3];
=0; i<3; i++)

=0; j<3; j++)

("%d", &arr[i][j]);


= 0; even = 0;
= 0; i<3; i++)

int j=0; j<3; j++)

((i+j)%2!=0)
odd += arr[i][j];
else
  even += arr[i][j];


tf ("%d\n%d", even, odd);
```

Program:

```
#include<stdio.h>
int main(){
    int i,j,x1,x2,n,y1,y2,t=0;
    long long total=0;
    int arr[1001][1001]={0};
    scanf("%d",&n);
    while(n--)
    {
      scanf("%d%d%d%d%d",&x1,&y1,&x2,&y2,&t);
      for(i=x1;i<=x2;i++)
      {
        for(j=y1;j<=y2;j++);
        {
          if arr[i][j]==0
          arr[i][j]+=t;
          else if(arr[i][j]>0)
          arr[i][j]=(-1)*(arr[i][j]++);
          else if(arr[i][j]<0){
          arr[i][j]==t;
        }
      }
    }
    for(i=1;i<1001;i++)
    {
      for(j=1;j<1001;j++)
      {if(arr[i][j]<0)
        total+=arr[i][j];
      }
    }
    printf("%lld\n",(-1)*total);
    return 0;
}
```

```
        printf("%d ", a[i].tal);
    }
```

Result:

The code is implemented and executed successfully.

Program:

```
#include<stdio.h>
int main() struct data{
{       int gen; int tal;
    };
int main(){
        int n;
        scanf("%d",&n);
        struct data a[n];
        for(int i=0, i<n;i++);{
            scanf("%d %d", &a[i].gen,&a[i].tal);
            for(int i=0, i<n-1; ;i++)
            {   for(int j=0,j<n-1-i;++j)
                {   if(a[j].tal < a[j+1].tal)
                    {
                        struct data temp = a[j];
                        a[j]=a[j+1];
                        a[j+1]= temp;
                    }
                }
            }
            for (int i=0; i<n;i++)
            {   if a[i].gen==0)
                printf("%d", a[i].tal);
                }
            for(int i=0; i<n;++i)
```

Character Arrays

gram:

```
include <stdio.h>
nt main() {
    char str1[10], str2[10], t;
    int i=0, j=0;
    int count1=0, count2=0;
    scanf("%s %s", str1, str2);
    while(str1[i] != '\0')
    {
        count1++
        i++;
    }
    while (str2[j] != '\0')
    {
        count2++;
        j++;
    }
    printf("%d %d\n", count1, count2);
    printf("%s %s\n", str1, str2);
    t = str1[0]
    str1[0] = str2[0];
    str2[0] = t;
    printf("%s %s", str1, str2);
    return 0;
}
```

ogram:

```
#include<stdio.h>
int main()
{
    char s[1000];
    scanf("%[^\n]s", s);
    for(int i=0; s[i]!='\0'; i++)
    {
        if(s[i]!=" ")
        printf("%c", s[i]);
        else
        printf("\n");
    }
    return 0;
}
```

```
:
clude <stdio.h>
main()

nar str[1000];
canf("%.s", str);
int hash[10] = {0,0,0,0,0,0,0,0,0,0}
int temp;
for(int i=0; str[i]=0';
if for(int i=0; str[i]!='\0'; i++)
{  temp = str[i]-'0'
   if (temp<=9 && temp>=0)
   {  hash[temp]++;
   }
   for(int i=0; i<=9; i++)
   {  printf("%.d", hash[i]);
   }
   return 0;
}
```

Program:

```
#include<stdio.h>
int main(){
    int t;
    scanf("%d",&t);
    while(t--){
        char str[100000];
        int count=0;
        scanf("%s",str);
        for(int i=0;str[i]!='\0';i++)
        {
            char c=str[i];
            if((c=='a')||(c=='e')||(c=='i')||
            (c=='o')||(c=='u')||(c=='A')||(c=='E')||
            (c=='I')||(c=='O')||(c=='U'))
            count++;
        }
        printf("%d\n",count);
    }
    return 0;
}
```

RESULT: The code is implemented and executed successfully.

Program:

```
#include<stdio.h>
int main(){
    int t;
    scanf("%d",&t);
    while(t--){
        char str[100000];
        int count=0;
        scanf("%s",str);
        for(int i=0;str[i]!='\0';i++)
        {
            char c=str[i];
            if((c=='a')||(c=='e')||(c=='i')||
            (c=='o')||(c=='u')||(c=='A')||(c=='E')||
            (c=='I')||(c=='O')||(c=='U'))
            count++;
        }
        printf("%d\n",count);
    }
    return 0;
}
```

RESULT: The code is implemented and executed successfully.

String Handling Function

String Handling Function

```
        flag = 0;
    }
    if (flag == 1)
    printf ("YES\n");
    }
    else {
       printf ("No\n");
    }
    return 0;
}
```

Program:

```
#include <stdio.h>
#include <string.h>
int main()
{   int t;
    scanf("%d", &t);
    while (t--)
    { int (flag = 1);
      char s[100000];
      scanf ("%s", s);
      int k = strlen(s);
      if (k==10)
      { for (int i = 0; i<10; i++) {
          { if (s[0] == '0')
            { flag = 0;
              break;
            }
            if (s[i]<'0' || s[i] > 'a')
             { flag = 0;
               break;
             }
          }
      }
```

```
else
flag = 0;
if (flag == 0)
Printf ("NO");
else
printf ("YES");
return 0;
}
```


Program:

```
#include<stdio.h>
#include<string.h>
int main()
{
  char str1[1000000], str2[1000000];
  int flag = 1;
  scanf("%s", str1);
  scanf("%s", str2);
  int a = strlen(str1);
  int b = strlen(str2);
  if (a == b)
  { for(int i=a-1; i>=0; i--)
    { while (str1[i] != str2[i])
      { for(int j=0; j<=i; j++)
        { if (str1[j] < 'z') {
          str1[j]++;
        else
        {
          flag 0;
          break;
        }
        if (flag == 0)
        break;
      }
    }
  }
}
```

Ex. No. :

Date :

Password

Problem Statement:

Danny has a possible list of passwords of Manny's facebook account. All passwords length is odd. But Danny knows that Manny is a big fan of palindromes. So, his password and reverse of his password both should be in the list.

You have to print the length of Manny's password and it's middle character.

Note: The solution will be unique.

Input Format
The first line of input contains the integer N, the number of possible passwords.
Each of the following N lines contains a single word, its length being an odd number greater than 2 and lesser than 14. All characters are lowercase letters of the English alphabet.

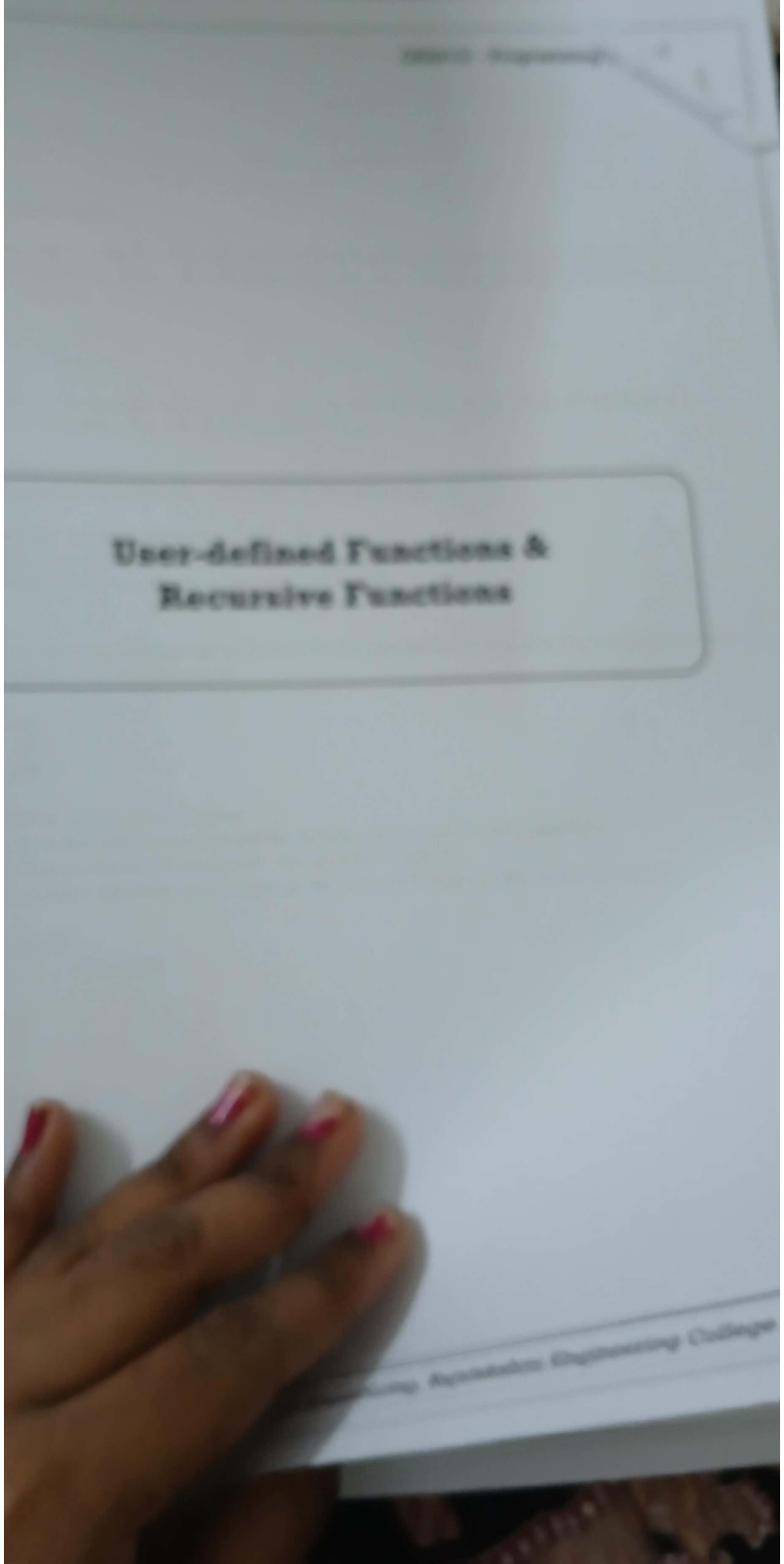
Output Format
The first and only line of output must contain the length of the correct password and its central letter.

Constraints
$1 \leq N \leq 100$

Sample Input
4
abc
def
feg
cba

Sample Output
3 b

[illegible]


User-defined Functions &
Recursive Functions

rogram:

```
#include <stdio.h>
nt main() {
    long pthFactor(long n, long p)
    {
      int count=0;
      for(long i=1; i<=n; ++i)
      {
        if(n%i==0)
        {
          count++;
          if(count==p)
          {
            return i;
          }
        }
      }
      return 0;
    }
```

```
de<stdio.h>
main() {
de<math.h>
rime (long long n) {
    if (n<=1) {
        return 0;
    }
    for (long long i= 2; i<= sqrt(n); i++) {
        if (n%i==0) {
            return i;
        }
    }
    return i;
  }
  return 1;
}
int main() {
        long long n;
```

```
int fit (int number)
{
  int binary[32],
  int i=0,
  while (number >0)
  {
    binary[i] = number % 2;
    number = number / 2;
    i++;
  }
  if (i>=4)
  {
    return binary[3];
  }
  else
    return 0;
  }
```

```
ude <stdio.h>
ude <math.h>
powersum (int x, int N, int num)

int power = pow(num, N);
if (power == x)
    return 1;
 if (power > x)
    return 0;
 return powersum (x - power, N, num+1)+
                powersum (x, N, num+1); }
int main(){
        int x, N;
        printf("
        scanf ("%d", &x);
        scanf ("%d", &N);
        printf ("%d\n", powersum (x, N, 1
        return 0;
      }
```

Ex. No. :

Date :

Hack the Money

Problem Statement:

You are a bank account hacker. Initially you have 1 rupee in your ac
exactly N rupees in your account. You wrote two hacks, first hack can
of money you own by 10, while the second can multiply it by 20. Thes
any number of time. Can you achieve the desired amount N using the

Constraints:

1<=T<=100
1<=N<=10^12

Input

- The test case contains a single integer N.

Output
For each test case, print a single line containing the string "1" if yo
rupees or "0" otherwise.

SAMPLE INPUT
1

SAMPLE OUTPUT
1

SAMPLE INPUT
2

SAMPLE OUTPUT
0

ram:

```
nclude <stdio.h>
t canhack (long long current , long long target)
 if ( current > target)
    return 0;
 if ( current == target)
     return 1;
  return  canhack ( current* 10, target) ||
            canhack (current * 20, target);
    }
int main(){
      int T;
      scanf ("%d", &T);
       while(T--) {
            long long N;
            scanf ("%lld", &N);
            if ( canhack (1, N)) {
              printf ("1\n";
             else.
              printf ("0\n");
       }
      return 0;
   }
```

```
de <stdio.h>
nhack (long long current, long long target)
( current > target)
 return 0;
 (current == target)
   return 1;
return canhack(current*10, target) ||
        canhack(current*20, target);
  }
int main(){
    int T;
    scanf("%d", &T);
    while(T--) {
        long long N;
        scanf("%lld", &N);
        if(canhack(1, N)) {
          printf("1\n");
        else
          printf("0\n");
    }
    return 0;
}
```

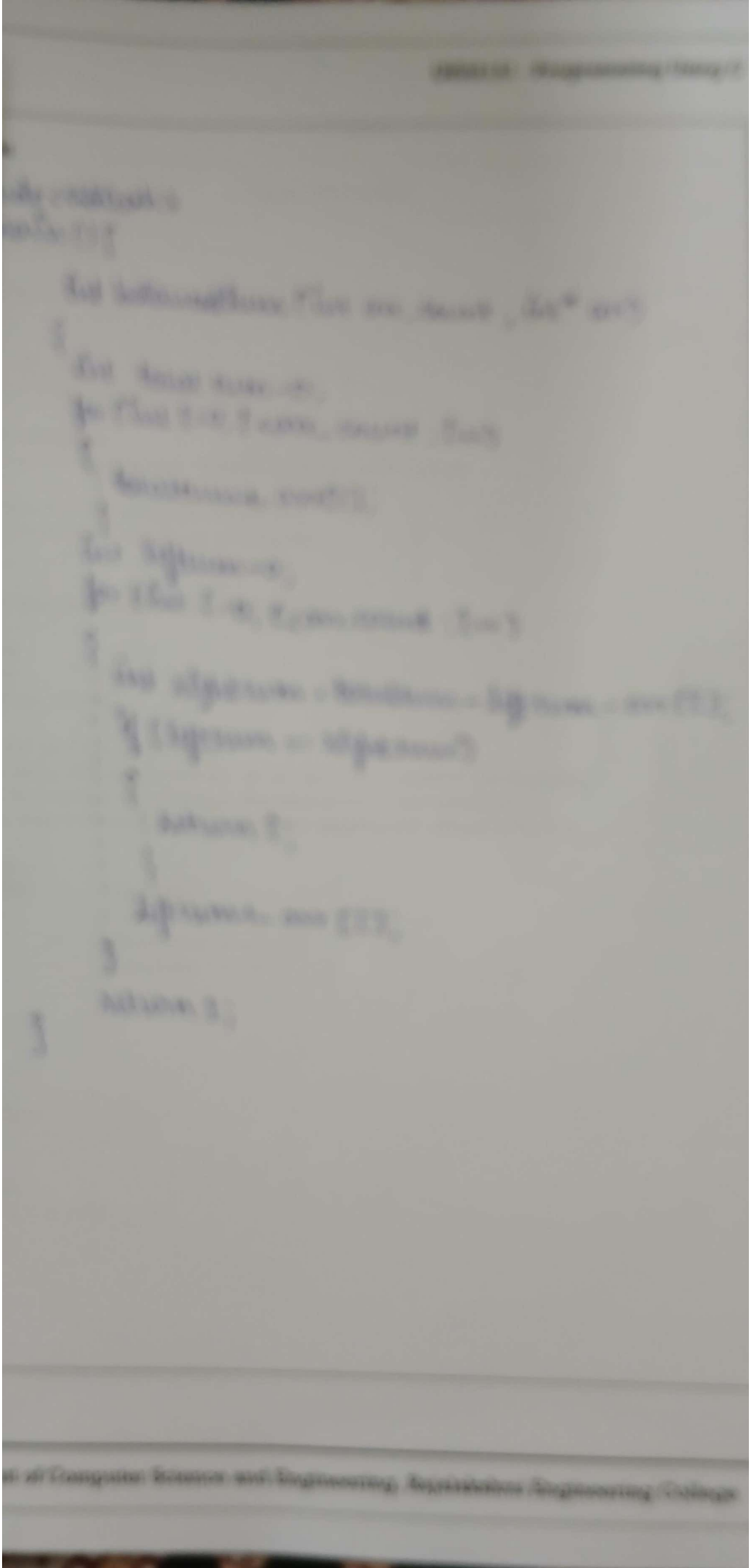
Program:

```
#include<stdio.h>
int main()
int largest divisor not divisible by k (int num, int k)
{
    for (int i = num-1 ; i>=1 ; i--){
        if (num%i == 0 && i%k != 0){
            return i;
        }
    }
    return 0;
}
int main (){
        int n, k;
        scanf ("%d %d", &n, &k);
        long long sum = 0
        for (int i = 1; i<= n; i++){
            sum = largest divisor not divisible by k(i
        }
        printf ("%d", sum);
        return 0;
}
```

```
ude <stdio.h>
ude <string.h>
uperdigit (long long num) {
  if (num<10) {
        return num;
    }
long long sum = 0;
 while (num>0) {
       sum += num%10;
        num1=10;
    }
     return superdigit (sum);
   }
int main(){
 Char n[10000001]
 int k;
 scanf("%s%d", n, &k);
 long long sum = 0;
 for(int i=0; i<strlen(n); i++) {
          sum += n[i] - '0';
    }
    sum *= k;
    int result = super digit(sum);
    printf ("%d\n",result);
    return 0;
}
```

RESULT: Hence the code is implemented & executed successfully.

Passing Arrays and Strings to Functions



Program:

```
#include<stdio.h>
int main(){

    int balancedSum(int arr_count, int* arr)
{
    int totalsum=0;
    for(int i=0; i<arr_count; i++)
    {
      totalsum+= arr[i];
    }
    int leftsum=0;
    for(int i=0; i<arr_count; i++)
    {
      int rightsum = totalsum - leftsum - arr[i];
      if(leftsum == rightsum)
      {
        return i;
      }
      leftsum+= arr[i];
    }
    return 1;
}
```

Program:

```
#include<stdio.h>
int main()
{
   int arraysum(int numbers_count,int *numbers)
   {
      int sum=0;
      for(int i=0;i<numbers_count;i++)
      {
         sum=sum+numbers[i];
      }
      return sum;
   }
}
```

Program:

```
#include<stdlib.h>
int main ()
{int compare (const void *a, const void *b)
 {
   return (*(int*)a - *(int *)b);
 }
int mindiff (int arr_count, int *arr)
{
   qsort (arr, arr_count, sizeof (int), compare);
    int totaldiff = 0;
    for (int i = 1; i<arr_count; i++)
    {
      total diff += abs (arr[i] - arr[i-1]);
    }
    return totaldiff;
}
```

Structures and Unions

Program:

```
#include<stdio.h>
int main()
{
    int n;
    scanf("%d",&n);
    for(int i=0; i<n; i++){
    {
        int length, width, h;
        scanf("%d %d %d", &length, &width, &h);
        if (h<41)
        {
            int volume = length*width*h;
            printf("%d\n", volume);
        }
    }
}
```

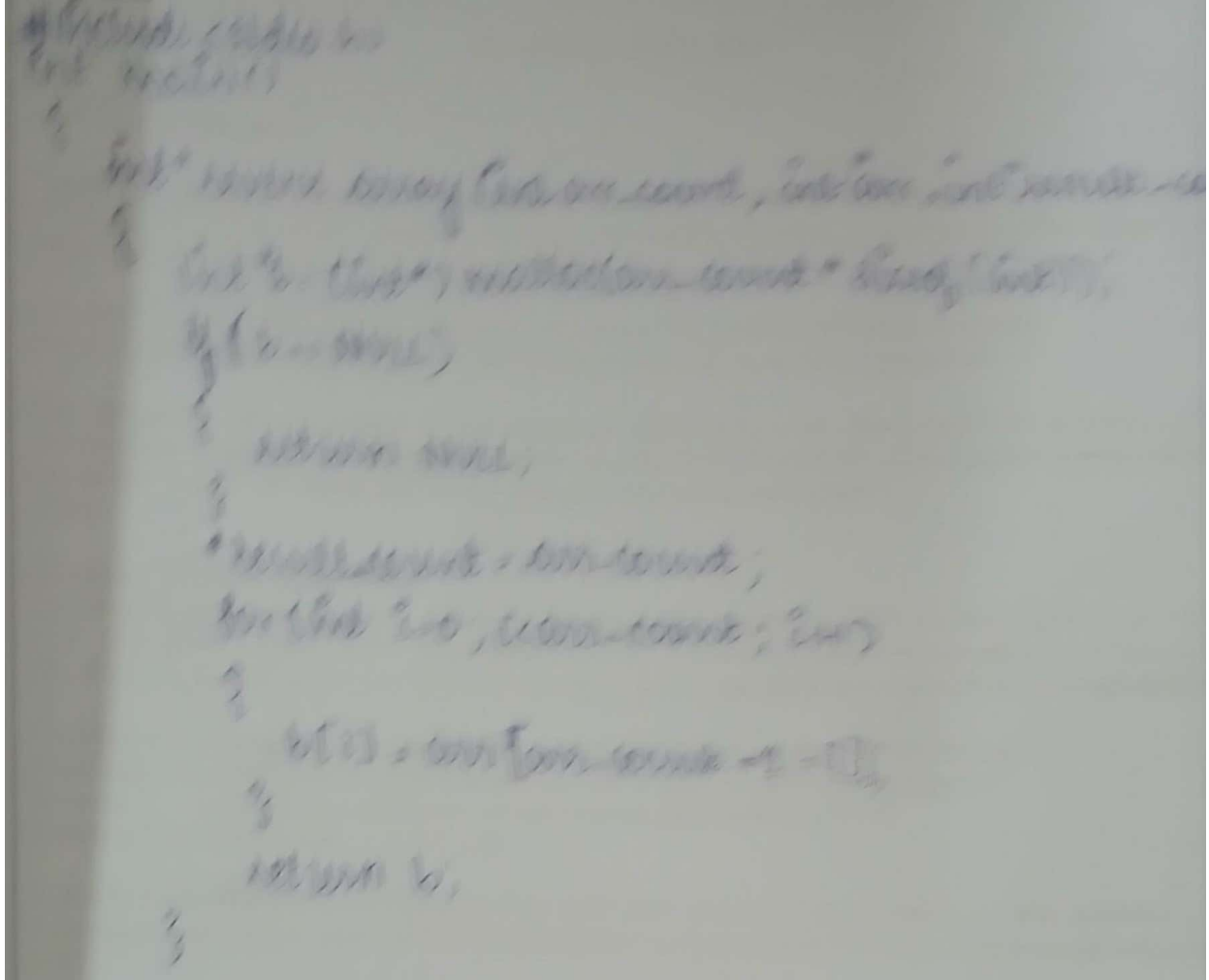
```
int main a, b, c;
scanf ("%d %d %d", &a, &b, &c);
triangles[i].a = a;
triangles[i].b = b;
triangles[i].c = c;
triangles[i].area = calculate_area(a,b,c);
}
qsort (triangles, n, sizeof (triangle), compare);
for (int i=0; i<n; i++)
{
  printf ("%d %d %d\n", triangles[i].a, triangles[i].
                                        triangles[i].c
}
return 0;
}
```


Program:

```
#include<stdio.h>
#include<math.h>
#include<stdlib.h>
typedef struct {
                double area;
                int a,b,c;
               }
        triangle;
double calculate_area (int a, int b, int c);
{
 double p=(a+b+c)/2.0;
 return sqrt (p*(p-a)*(p-b)*(p-c));
}
int compare (const void *x, const *void y)
{
 triangle *t1 = (triangle*) x;
 triangle *t2 = (triangle*) y;
 if (t1->area < t2->area) return -1;
 if (t1->area > t2->area) return 1;
 return 0;
}
int main (){
            int n;
            scanf ("%d", &n);
            triangle triangles [n];
            for (int i=0; i<n; i++)
            {
```

Pointers

Program:



Program:

```
#include<stdio.h>
int main()
{
    char* cutthemall(int lengths_count, long *lengths,
                                        long minlength)
    {
        long totallength = 0;
        for(int i=0; i<lengths_count; i++)
        {
            totallength += lengths[i];
        }
        if(totallength < minlength)
        {
            return "Impossible";
        }
        long remaininglength = totallength;
        for(int i= 0; i < lengthscount; i++)
        {
            remaining length -= lengths[i];
            if(remaining length >= min length)
            {
                return "Possible";
            }
        }
        return "Impossible";
    }
}
```

Program:

```
char* cut thewall (int lengths_count, long *lengths, long min length)
{
  long total length = 0;
  for (int i=0, i<lengths_count ; i++){
    total length += lengths [i];
  }
  if (total length< min length)
  {
    return "Impossible";
  }
  long remaining length = total length;
  for (int i = 0; i< lengths_counts ; i++ )
  {
    remaining length -= lengths [i];
    if (remaining length >=minlength)
    {
      return "Possibe";
    }
  }
  return 'Impossible"
}
```

Program:

```
char* cut themall (int lengths_count, long *lengths, long min length)
{
  long total length = 0;
  for (int i=0, i<lengths_count ; i++){
     total length += lengths [i];
  }
  if (total length < min length)
  {
    return "Impossible";
  }
  long remaining length = total length;
  for (int i = 0; i< lengths_counts ; i++ )
  {
    remaining length -= lengths [i];
    if (remaining length >=minlength)
    {
      return "Possibl";
    }

  }
  return 'Impossible"
}
```