Course: C **Session:** Structure Pointers and Array of Pointers **Timestamp:** 2021-1-9 23:03:02 **Register Number:** RA2031241010094

Q. Pointers - 10

Write a program to add two integers using functions use call by address technique of passing parameters and also illustrate the concept of pointer variables can be used to access the strings.

Input and Output Format:

Refer sample input and output for formatting specification.

All float values are displayed correct to 2 decimal places.

All text in bold corresponds to input and the rest corresponds to output.

Source Code

```
#include <stdio.h>
int main()
{
  int a,b,sum;
  scanf("%d %d",&a,&b);
  sum=res(a,b);
  printf("The sum of the numbers is %d\n",sum);
  printf("Accessing a string using pointer\n");
  printf("Hello");
 return (0);
}
int res(int a,int b)
{
  int c,*ptr=&a,*ptr1=&b;
  c=*ptr+*ptr1;
  return(c);
}
```

Sample Input

```
6
7
```

Sample Output

```
The sum of the numbers is 13
Accessing a string using pointer
Hello
```

Result

Thus, Program " **Pointers - 10** " has been successfully executed

Course: C **Session:** Structure Pointers and Array of Pointers **Timestamp:** 2021-1-9 23:03:40 **Register Number:** RA2031241010094

Q. Back 2 Back

Rand has a task to find the sum of first and last digit of number upto given n numbers.Help him in writing C code to enter any number and find the sum of first and last digit of the number using for loop.

Source Code

```
#include <stdio.h>
int main()
{
  int n,f,l,sum=0;
  scanf("%d",&n);
  l=n%10;
  f=n;
  while(n>=10)
  {
    n=n/10;
  }
  f=n;
  sum=f+l;
  printf("%d",sum);
 return 0;
}
```

Sample Input

```
1234
```

Sample Output

```
5
```

Result

Thus, Program " **Back 2 Back** " has been successfully executed

Q. Adding two distances

1. Create a Structure called "Distance"
2. Create two data members of "Distance Structure" feet(int), inch(float)
3. Create three structure variables as d1, d2 and sumOfDistances 4. Get two distances and add the feet and inches.

Mandatory:

To add the distance using the structure variables as follows

1. sumOfDistances.feet=d1.feet+d2.feet

2 sumOfDistances.inch=d1.inch+d2.inch

Source Code

```
#include <stdio.h>
struct Distance
{
  int feet;
  float inch;
}d1,d2,sumOfDistances;
int main()
{
  scanf("%d %f\n",&d1.feet,&d1.inch);
  scanf("%d %f\n",&d2.feet,&d2.inch);
  {
    sumOfDistances.feet=d1.feet+d2.feet;
    sumOfDistances.inch=d1.inch+d2.inch;
    printf("Sum of distances=%d feet and %.2f inches",sumOfDistances.feet,sumOfDistances.inch);
  }
 return 0;
}
```

Sample Input

```
23 8.6
34 2.4
```

Sample Output

```
Sum of distances=57 feet and 11.00 inches
```

Result

Thus, Program " **Adding two distances** " has been successfully executed

Course: C **Session:** Structure Pointers and Array of Pointers **Timestamp:** 2021-1-9 23:04:35 **Register Number:** RA2031241010094

Q. Magic Square

A magic square is an arrangement of numbers (usually integers) in a square grid, where the numbers in each row, and in each column, and the numbers in the forward and backward main diagonals, all add up to the same number

Input Format:

The input consists of (n*n+1) integers. The first integer corresponds to the number of rows/columns in the matrix. The remaining integers correspond to the elements in the matrix. The elements are read in rowwise order, first row first, then second row and so on. Assume that the maximum value of m and n is 5.

Source Code

```
#include <stdio.h>
int main()
{
 int size=3;
 int a[3][3];
 int i,j=0;
 int sum,sum1,sum2;
 int flag=0;
 for(i=0;i<size;i++)
 {
  for(j=0;j<size;j++)
   scanf("%d",&a[i][j]);
 }
 sum=0;
 for(i=0;i<size;i++)
 {
  for(j=0;j<size;j++)
  {
   if(i==j)
    sum=sum+a[i][j];
  }
 }
 for(i=0;i<size;i++)
 {
  sum1=0;
  for(j=0;j<size;j++)
  {
   sum1=sum1+a[i][j];
  }
  if(sum==sum1)
   flag=1;
  else
  {
   flag=0;
   break;
  }
 }
 for(i=0;i<size;i++)
 {
  sum2=0;
  for(j=0;j<size;j++)
  {
   sum2=sum2+a[j][i];
  }
  if(sum==sum2)
   flag=1;
  else
  {
   flag=0;
   break;
  }
 }
 if(flag==1)
  printf("Magic Square");
 else
  printf("Not a Magic Square");
 return 0;
}
```

Sample Input

```
4 9 2
3 5 7
9 3 5
```

Sample Output

```
Not a Magic Square
```

Result

Thus, Program " **Magic Square** " has been successfully executed

Course: C **Session:** Structure Pointers and Array of Pointers **Timestamp:** 2021-1-9 23:05:00 **Register Number:** RA2031241010094

Q. Even or odd

Somesh and sakthi played one game to find the number is even or not, for that they have designed one coding, using union concept track yourself by your code to challenge them for their inputs.

Input Method

Integer ranges from 1 to 999

Output Method

Print the number is even or not

Source Code

```
#include <stdio.h>
union eo
{
  int a;
}s;
int main()
{
  scanf("%d",&s.a);
  if(s.a%2==0)
    printf("Even");
  else
 printf("Odd");
 return 0;
}
```

Sample Input

```
2
```

Sample Output

```
Even
```

Result

Thus, Program " **Even or odd** " has been successfully executed

Q. Caravan

Most problems on CodeChef highlight chefs love for food and cooking but little is known about his love for racing sports. He is an avid Formula 1 fan. He went to watch this years Indian Grand Prix at New Delhi. He noticed that one segment of the circuit was a long straight road. It was impossible for a car to overtake other cars on this segment. Therefore, a car had to lower down its speed if there was a slower car in front of it. While watching the race, Chef started to wonder how many cars were moving at their maximum speed.

Formally, youre given the maximum speed of N cars in the order they entered the long straight segment of the circuit. Each car prefers to move at its maximum speed. If thats not possible because of the front car being slow, it might have to lower its speed. It still moves at the fastest possible speed while avoiding any collisions. For the purpose of this problem, you can assume that the straight segment is infinitely long.

Count the number of cars which were moving at their maximum speed on the straight segment.
Input

The first line of the input contains a single integer T denoting the number of test cases to follow. Description of each test case contains 2 lines. The first of these lines contain a single integer N, the number of cars. The second line contains N space separated integers, denoting the maximum speed of the cars in the order they entered the long straight segment.
Output

For each test case, output a single line containing the number of cars which were moving at their maximum speed on the segment.

Source Code

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

Sample Input

```
3
1
10
3
8 3 6
5
4 5 1 2 3
```

Sample Output

```
1
2
2
```

Result

Thus, Program " **Caravan** " has been successfully executed

Course: C **Session:** Structure Pointers and Array of Pointers **Timestamp:** 2021-1-9 23:05:52 **Register Number:** RA2031241010094

Q. Sum of Negative Numbers

Write a program to find the sum of negative numbers in an array.

Input Format:
Input consists of n+1 integers. The first integer corresponds to n , the size of the array. The next n integers correspond to the elements in the array. Assume that the maximum value of n is 15.

Output Format:
Refer sample output for details.

Source Code

```
#include <stdio.h>
int main()
{
  int a[50],n,i,t=0;
  scanf("%d",&n);
  for(i=0;i<n;i++)
  {
    scanf("%d",&a[i]);
    if(a[i]<0)
    {
      t=t+a[i];
    }
  }
 printf("sum=%d",t);
 return 0;
}
```

Sample Input

```
5
2 3 6 8 -1
```

Sample Output

```
sum=-1
```

Result

Thus, Program " **Sum of Negative Numbers** " has been successfully executed

Course: C **Session:** Structure Pointers and Array of Pointers **Timestamp:** 2021-1-9 23:06:19 **Register Number:** RA2031241010094

Q. Lucy

Lucy is celebrating her 15th birthday, Her father promised her that he will buy her a new computer on her birthday if she solves the question asked by him. He asks Lucy to find whether the year on which she had born is leap year or not. Help her to solve this puzzle so that she celebrates her birthday happily. If her birth year is 2016 and it is a leap year display 2016 is a leap year.? Else display 2016 is not a leap year and check with other leap year conditions.

Source Code

```
#include <stdio.h>
int main()
{
 int year;
 scanf("%d",&year);
 if(year%400==0)
 {
   printf("%d is a leap year",year);
 }
 else if(year%100==0)
 {
   printf("%d is not a leap year",year);
 }
 else if(year%4==0)
 {
   printf("%d is a leap year",year);
 }
 else
 {
printf("%d is not a leap year",year);
 }
return 0;
}
```

Sample Input

```
1900
```

Sample Output

```
1900 is not a leap year
```

Result

Thus, Program " **Lucy** " has been successfully executed

Course: C **Session:** Structure Pointers and Array of Pointers **Timestamp:** 2021-1-9 23:06:45 **Register Number:** RA2031241010094

Q. Semester Holidays

Normally in all engineering colleges, there will be long vacation after every even semester and a short vacation after every odd semester.

Input format:

Input consists of 1 integers which corresponds to the current semester of the students (i.e) Even semester " Long Vacation"
ODD semester "Short Vacation" determine by dividing(modulo) with 2

Output format:

Output consists of the string "Long Vacation " or "Short Vacation".

Refer sample input and output for further formatting specifications.

Source Code

```
#include <stdio.h>
int main()
{
  int n;
  scanf("%d",&n);
  if(n%2==0)
    printf("Long Vacation");
  else
 printf("Short Vacation");
 return 0;
}
```

Sample Input

```
6
```

Sample Output

```
Long Vacation
```

Result

Thus, Program " **Semester Holidays** " has been successfully executed

Course: C **Session:** Structure Pointers and Array of Pointers **Timestamp:** 2021-1-9 23:07:15 **Register Number:** RA2031241010094

Q. Rectangular Object

Subbu needs to make a rectangular box for his physics class project. He has bought P cm of wire and S cm2 of special paper. He would like to use all the wire (for the 12 edges) and paper (for the 6 sides) to make the box.

What is the largest volume of the box that Johnny can make?
Input

The first line contains t, the number of test cases (about 10). Then t test cases follow.

Each test case contains two integers P and S in a line (1<= P <= 40000, 1 <= S <= 20000). You may assume that there always exists an optimal solution for the given input cases.
Output

For each test case, print a real number that is the largest volume of the box that Johnny can make, rounded to two decimal places.

Source Code

```
#include <stdio.h>
#include <math.h>
int main()
{
  int t,p,s;
  float v,h,d;
  scanf("%d",&t);
  while(t--)
  {
    scanf("%d %d",&p,&s);
    d=sqrt(p*p-24*s);
    h=(p-d)/12;
    v=h*h*h-((h*h*p)/4)+((s*h)/2);
 printf("%.2f\n",v);
  }
  return 0;
}
```

Sample Input

```
2
20 14
20 16
```

Sample Output

```
3.00
4.15
```

Result

Thus, Program " **Rectangular Object** " has been successfully executed