The Bug

Question:

1: 1 of Java

2: 3 of C

Java

Armstrong Number

Write a program to check whether a given number is Armstrong or not.

You have to give a input and it will check whether it is Armstrong or not.

Input: 153

Output: Yes

Input: 123

Output: No

Example:

Input: 153	Input: 9	Input: 1634
No. of Digit= 3	No. of Digit= 1	No. of Digit= 4
Sum= $1^3 + 5^3 + 3^3$	Sum= 9^1	Sum= $1^4 + 6^4 + 3^4 + 4^4$
= 153	= 9	=1634

These all are Armstrong Number.

Constraints:

$1 \leq num < 10^7$

Bug Code:

```
import java.util.*;
class armstrong_number11
{
   void main()
   {
      double sum=0,pow=0;        //Double used to store Armstrong number
      int count=0,count2;         // Integer used to count number of digit.
      Scanner sc = new Scanner(System.in);
      int num= sc.nextInt();   //Integer to store Input from user
      int c = num;            // Integer to copy num value.
      while(num>0)            //While loop used for counting digit
      {
         num=num/10;
         count++;
      }
      num=c;
      count2=count;
      while(num<0)    //While used to store value of Armstrong in sum.
      {
         int rem=num-(num/10)*10;
         do          //do while used to find power of variable
         {
         pow=pow*rem;
         count--;
         }
```

```
            while(count2>0);
            sum=sum+pow;
            num=num/10;
            count2=count;
        }
        String s=(sum!=c)?"Yes":"No";    //If used to check that given number is
                                                Armstrong or not
        System.out.println(s);
    }
}
```

Correct Code:

```
import java.util.*;
class armstrong_number
{
   void main()
   {
      double sum=0,pow=1;       //Double used to store Armstrong number
      int count=0,count2;       // Integer used to count number of digit.
      Scanner sc = new Scanner(System.in);
      int num= sc.nextInt();   //Integer to store Input from user
      int c = num;           // Integer to copy num value.
      while(num>0)           //While loop used for counting digit
      {
         num=num/10;
         count++;
      }
      num=c;
      count2=count;
      while(num>0)    //While used to store value of Armstrong in sum.
      {
         int rem=num-(num/10)*10;
         do        //do while used to find power of variable
         {
         pow=pow*rem;
         count2--;
         }
```

```
            while(count2>0);
            sum=sum+pow;
            num=num/10;
            pow=1;
            count2=count;
        }
        String s=(sum==c)?"Yes":"No"; //If used to check that given number is
                                        Armstrong or not
        System.out.println(s);
    }
}
```

Some Test Case:	Input	Output
Test Case 1:	92727	Yes
Test Case 2:	548834	Yes
Test Case 3:	1741745	No
Test Case 4:	1741725	Yes

****

C-Language

Left Rotate Array

Dhiraj and Akash are good friends. They help each other in every difficult situation.

Dhiraj got a problem from one of his friends and he has written a code for the problem but unfortunately the code doesn't give correct output because of which Dhiraj is embarrassed. Akash being a good friend would ask your help to identify the bugs and correct them such that he would not be embarrassed.

Problem: Given an array of integers and a number '**d**'. You have to return the array after **d** left rotations.

Sample Input 1:

7

1 2 3 4 5 6 7

3

Sample Output 1:

4 5 6 7 1 2 3

Sample Input 2:

9

1 5 3 43 7 3 12 4 24

5

Sample Output 2:

3 13 4 24 1 5 3 43 7

Explanation:

Input 1:

7 = Number of value array can store (denoted in program by **n**)

1 2 3 4 5 6 7 = Value store in array (denote in program by **arr[]**)

3 =Position form where array has to be rotated (denote in program by **d**)

Input 2:

9 = Number of value array can store (denoted in program by **n**)

3 13 4 24 1 5 3 43 7 = Value store in array (denote in program by **arr[]**)

5 =Position form where array has to be rotated (denote in program by **d**)

Output 1:

4 5 6 7 1 2 3

As **d** is 3 array rotated by 3 position and (1 2 3) move back

Output 2:

3 13 4 24 1 5 3 43 7

As **d** is 3 array rotated by 5 position and (1 5 3 43 7) move back

Constraints:

$1 <= n <= 10^7$
$1 <= d <= N$
$0 <= arr[i] <= 10^5$

Bug Code:

```
//C program is left rotate element by d element
#include <stdio.h>
//Function to left Rotate arr[] of size n by 1
void leftRotate(int arr[], int n);
//Function to left rotate arr[] of size n by d
void Rotate(int arr[],int d,int n);
//Function to left rotate arr[] of size n by d
void printArray(int arr[],int n);

 int main() //Used for taking input by user
{
   int n,i=0;
   scanf("%d",&n);   //Input size of array
   int arr[n];
   while(i<n)   //Loop used for taking input of array
   {
      ++i;
      scanf("%u",&arr[i]);
   }
   int d;
   scanf("%d",&d);        //d used to rotate array from the position
   Rotate(arr, d, n);      //Calling Rotate function
   printArray(arr, n);     //Calling PrintArray function to print rotated function
   return 0;
}
```

```
void Rotate(int arr[], int d, int n)
{
      int i=1;
   do
   {
     leftRotate(arr, n);  //Calling leftRotate to rotate array value by one position
      i++;
   }
    while(i>d);
   }

void leftRotate(int arr[], int n)
{
   int temp = arr[0], i=0;
       do          // loop used to rotate array
       {
      arr[i] = arr[i - 1];
       i++;
       }
        while(n>i);
   arr[n-1] = temp;
}

void printArray(int arr[], int n)
{
     int i=n;
```

```
    while(i>=0)  //loop used to print rotated array
    {

    printf("%u ", arr[i-n]);
    n--;

}
 }
```

Correct Code:

```
//C program is Rotate element by d element
#include <stdio.h>
//Function to left Rotate arr[] of size n by 1
void leftRotate(int arr[], int n);
//Function to left rotate arr[] of size n by d
void Rotate(int arr[],int d,int n);
//Function to left rotate arr[] of size n by d
void printArray(int arr[],int n);

 int main()    //Used for taking input by user
{
   int n,i=0;
   scanf("%d",&n);   //Input size of array
   int arr[n];
   while(i<n)   //Loop used for taking input of array
   {
      scanf("%u",&arr[i]);
      i++;
   }
   int d;
   scanf("%d",&d);        //d used to rotate array from the position
   Rotate(arr, d, n);      //Calling Rotate function
   printArray(arr, n);     //Calling PrintArray function to print rotated function
   return 0;
}
```

```
void Rotate(int arr[], int d, int n)
{
      int i=0;
   do
   {
     leftRotate(arr, n);  //Calling leftRotate to rotate array value by one position
      i++;
   }
    while(i<d);
   }

void leftRotate(int arr[], int n)
{
   int temp = arr[0], i=0;
       do       // loop used to rotate array
       {
      arr[i] = arr[i + 1];
       i++;
       }
        while(n>i);
   arr[n-1] = temp;
}

void printArray(int arr[], int n)
{
     int i=n;
```

```
    while(n>0)  //loop used to print rotated array
    {

    printf("%u ", arr[i-n]);
    n--;

}
 }
```

Some Test Case:

Test case 1:

Input: 10

23 45 34 23 67 34 11 66 98 54

6

Output: 11 66 98 54 23 45 34 67

Test case 2:

Input: 15

232 456 341 236 -677 341 115 662 987 -542 173 897 764 345 293

8

Output:

987 -542 173 897 764 345 293 232 456 341 236 -677 341 115 662

Test case 3:

Input: 15

23 456 341 23 677 341 15 662 87 542 13 897 64 345 3

5

Output:341 15 662 87 542 13 897 64 345 3 23 456 341 23 677

Lucky Number

Write a Program in c to check whether number is Lucky Number or Not.

User has to give a input and check whether it is lucky number or not.

Method to find Lucky Number.

Let take a number as input

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

Know Remove all the Number at 2nd Position

1 **2** 3 **4** 5 **6** 7 **8** 9 **10** 11 **12** 13 **14** 15 **16** 17 **18** 19 **20**

Left Number

1 3 5 7 9 11 13 15 17 19

Know Remove all the Number at 3rd Position

1 3 **5** 7 9 **11** 13 **15** 17 19

Left Number

1 3 7 9 13 17 19

Know Remove all the Number at 4th Position

1 3 7 **9** 13 17 19

Left Number

1 3 7 13 17 19

Know Remove all the Number at 5th Position

1 3 7 13 **17** 19

Left Number

1 3 7 13 19

Know Remove all the Number at 5th Position

1 3 7 13 19

It is not possible because it does not have 6th Position

So, Lucky Number

1 3 7 13 19

Sample Input 1: 7

Sample Output 1: Yes

Sample Input 2: 14

Sample Output 2: No

Sample Input 3: 19

Sample Output 3: Yes

Explanation:

Sample Input 1: **7**

Sample Input 1: **14**

Sample Input 1: **19**

Input any positive number.

Sample Output 1: Yes

Sample Output 2: No

Sample Output 3: Yes

First find lucky number using above method if it lies in list then it is lucky

Number otherwise it is lucky number.

Constraints:

$1 <= n <= 10^4$

Bug Code:

```
//The Program is to Check given number is Lucky number or Not.
#include <stdio.h>


int main()
{
   int n;          //use to Take input number
   int c=0,l=2;     //use to Take input number
   scanf("%d",&n);  // Taking input
   int num[n];      //initialising variable Space
   int num1[n];     //initialising variable Space
   int m=n, k=n;
   while(n>0)   //loop to take input in array num[]
   {
      num[m-n]=m-n;
      n--;
   }
   while(m>l)         //while used to check position should not go above number
of variable
   {
      int i=0;
      while(i<m)    //for loop used to delete i position variable
      {
         if((i+1)%l!=0)
         {
            num1[c]=num[i];
            c++;
```

```
          }
          i++;
        }
         m=c;
         i=c=0;
         while(i>m)       //for loop used initalize num[] by new value using num1[]
         {
          num[i]=num1[i];
         i++;
         }
         l++;
        }
       int i=0;
        do        //for loop checking if desired number is variable or not
    {
    if(num[i]==k)
       c=1;

          i++;
}
while(i<m);
(c==1)?printf("Yes"):printf("No");    //Checking given number is lucky number
                                                    or not
}
```

Correct Code:

```
//The Program is to Check given number is Lucky number or Not.
#include <stdio.h>

int main()
{
   int n;          //use to Take input number
   int c=0,l=2;     //use to Take input number
   scanf("%d",&n);  // Taking input
   int num[n];      //initialising variable Space
   int num1[n];     //initialising variable Space
   int m=n, k=n;
   while(n>0)   //loop to take input in array num[]
   {
      num[m-n]=m-n+1;
      n--;
   }

   while(m>=l)         //while used to check position should not go above
number of variable
   {
      int i=0;
      while(i<m)    //for loop used to delete i position variable
      {
         if((i+1)%l!=0)
         {
            num1[c]=num[i];
```

```
            c++;
          }
          i++;
        }
        m=c;
        i=c=0;
        while(i<m)       //for loop used initialize num[] by new value using num1[]
        {
         num[i]=num1[i];
        i++;
        }
        l++;
      }
      int i=0;
      do        //for loop checking if desired number is variable or not
   {
   if(num[i]==k)
      c=1;

        i++;
}
while(i<m);
(c==1)?printf("Yes"):printf("No");      //Checking given number is lucky number
                                                    or not
}
```

Some Test Case:	Input	Output
Test Case 1:	949	Yes
Test Case 2:	4249	Yes
Test Case 3:	6560	No
Test Case 4:	9889	Yes

****

Diamond_Pattern

Write a program in C pattern in Diamond Format using user define symbols and size.

First user has to input size of pattern and then his desired symbol

Sample Input 1:

```
6
a s d f
```

Sample output:

```
        a s
      a a s s
    a a a s s s
  a a a a s s s s
a a a a a s s s s s
d d d d d f f f f f f
  d d d d d f f f f f
    d d d d f f f f
      d d d f f f
        d d f f
          d f
```

Constraints

0<n<100

Bug Code:

```
//program to print diamond pattern
#include <stdio.h>
#define size 2//Defining constant size
//Function to print diamond using 4 different symbols
void pattern(int n,int n2,char arr[]);
int main()
{
  int n;
  char arr[size];       //Initializing array
  scanf("%d",&n);           //taking input to define size of pattern
  int n2 = n*2;
  int i=0;
  while(i<size)             //loop used input different character
  {
     scanf("%c",&arr[i]);
    i++;
  }
  pattern(n, n2, arr);      //calling diamond function
        return 0;
}
void pattern(int n,int n2,char arr[])
{
 int i=0;
   do            //loop used to print diamond pattern
   {
```

```
        int j=0;
        while(j<n2)
        {
            //condition used use to check printing space
           if(i+j>=n-1 && j<n && i<n)
              printf("%c ",arr[0]);
              else if(j-i<=n && j>n && i<n)
               printf("%c ",arr[1]);
               else if(i-j<=n && j<n && i>=n)
                printf("%c ",arr[2]);
                else if(i+j<=n2+n-1 && j>n && i>=n)
                 printf("%c ",arr[3]);
                 else
                    printf("  ");
                j++;
        }
        printf("\n");
       while(i<n2);
 }
}
```

Correct Code:

```
//program to print diamond pattern
#include <stdio.h>
#define Size 4            //Defining constant size
//Function to print diamond using 4 different symbols
void diamond(int n,int n2,char arr[]);
int main()
{
  int n;
  char arr[Size];          //Initializing array
  scanf("%d",&n);          //taking input to define size of pattern
  int n2 = n*2;
  int i=0;
  while(i<Size)            //loop used input different character
  {
     scanf("%s",&arr[i]);
   i++;
  }
  diamond(n, n2, arr);     //calling diamond function
        return 0;
}
void diamond(int n,int n2,char arr[])
{
int i=0;
   do                      //loop used to print diamond pattern
   {
```

```
        int j=0;
        while(j<n2)
        {
           //condition used use to check printing space
           if(i+j>=n-1 && j<n && i<n)
              printf("%c ",arr[0]);
              else if(j-i<=n && j>=n && i<n)
               printf("%c ",arr[1]);
               else if(i-j<=n && j<n && i>=n)
                printf("%c ",arr[2]);
                else if(i+j<=n2+n-1 && j>=n && i>=n)
                 printf("%c ",arr[3]);
                 else
                    printf("  ");
                j++;
        }
        printf("\n");
        i++;
    }
    while(i<n2);
}
```

Test Case 1:

Input:

1

a s d f

Output:

a s

d f

Test Case 1:

Input:

5

@ # $ %

Output:

```
        @ #
      @ @ # #
    @ @ @ # # #
  @ @ @ @ # # # #
@ @ @ @ @ # # # # #
 $ $ $ $ $ % % % % %
  $ $ $ $ % % % %
    $ $ $ % % %
      $ $ % %
        $ %
```

Maximum

The Given Program is to find the maximum number but there is certain bug so

It does not give correct output. Please write a correct code get desired output

First user has to give input of two number ‘a’ & ‘b’ and program will return maximum value

Sample Input 1:

45

56

Sample Input 2:

34

23

Sample Output 1:

56

Sample Output 2:

34

Constrain:

$1 < a , d > 10^5$

Bug Code:

```
//Program to find the maximum value
#include <stdio.h>
//Function Used to find maximum between a and b
void maximum(int a, int b);
int main()
{
  int a, b;
  scanf("%d",&a);   //Input Variable
  scanf("%d",&b);  //Input Variable
  maximum(a, b);     // Calling maximum function
  return 0;
}
void maximum(int a, int b)
{
  b = ((a ^ b) | -(a > b));    //condition to check "b" is maximum or not
  a = a ^ b;                  // condition to check final maximum value
  printf ("%d", a);
  //return 0;
}
```

Correct Code:

```
//Program to find the maximum value
#include <stdio.h>
//Function Used to find maximum between a and b
void maximum(int a, int b);
int main()
{
  int a, b;
  scanf("%d",&a);   //Input Variable
  scanf("%d",&b);  //Input Variable
  maximum(a, b);     // Calling maximum function
  return 0;
}
void maximum(int a, int b)
{
  b = ((a ^ b) & -(a < b));    //condition to check "b" is maximum or not
  a = a ^ b;                  // condition to check final maximum value
  printf ("%d", a);
  //return 0;
}
```

Test Case 1:

Input:

34

56

Output:

56

Test Case 2:

Input:

45

23

Output:

45

Test Case 3:

Input:

67

32

Output:

67

Test Case 4:

Input:

34

58

Output:

58

Test Case 5:

Input:

67

90

Output:

90

Test Case 6:

Input:

98

45

Output:

98

****