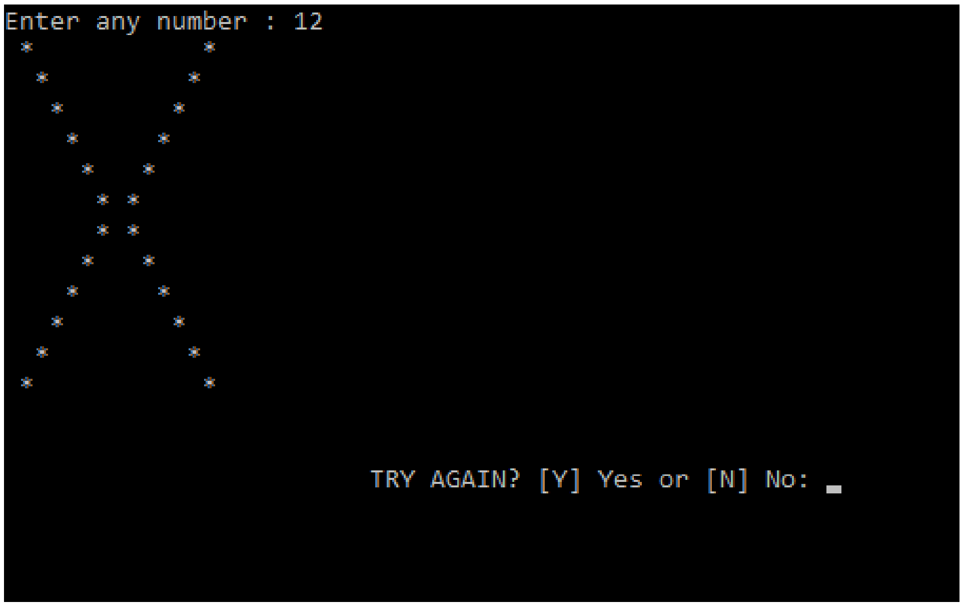
Program #1:
Source Code:

```
PrimaryMission() {
unit i,j,n;
unit temp;
unit choice=0;
go {
commence;
post("Enter any number : ");
capture(#n);
phase(n <= 0) {
post("Enter POSITIVE number : ");
capture(#n);
}
inquire(i=0;i<n;i++) {
inquire(j=0;j<n;j++) {
temp = n - i -1;
inorder((i==j) || (j==temp)) {
post(" *");
}
order {
post(" ");
}
}
post("\n");
}
go {
company ch;
post("\n\n\t\t\tTRY AGAIN? [Y] Yes or [N]
No: ");
capture(#ch);
inorder((ch = = "Y") || (ch = = "y")) {
i=0;
j=0;
n=0;
temp=0;
choice = 1;
}
otherorder((ch = = "N") || (ch = = "n")) {
choice = 0;
}
order {
post("\n\t\t\tError Input!");
choice = 3;
}
} phase(choice = = 3);
} phase(choice != 0);
post("\n\t\t\tGOODBYE!!");
} deploy();
```

Print Screen:

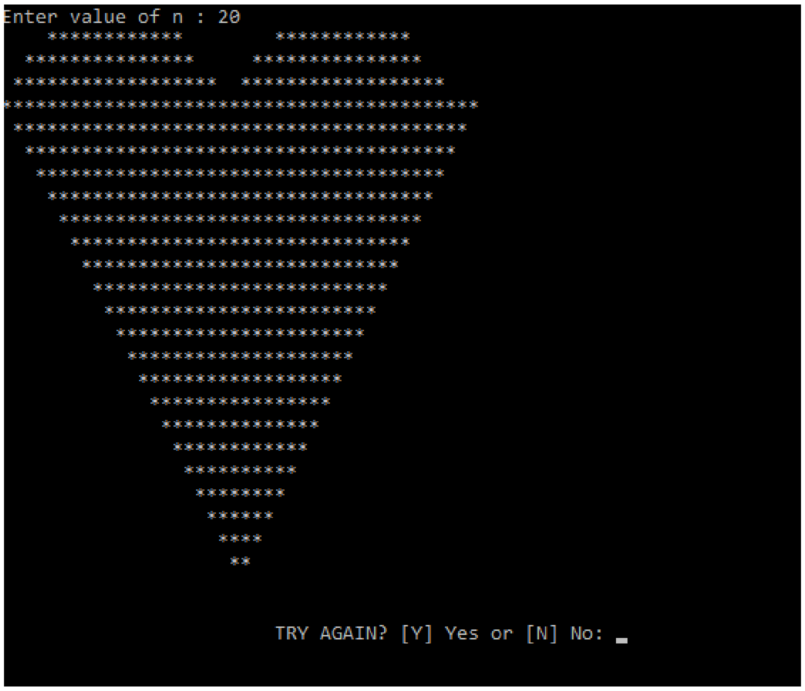


```
Enter any number : 12
 *           *
  *         *
   *       *
    *     *
     *   *
      * *
      * *
     *   *
    *     *
   *       *
  *         *
 *           *


                        TRY AGAIN? [Y] Yes or [N] No:
```

Program # 2:
Source Code:

```
PrimaryMission() {
unit i, j, n,k,temp,temp2,temp3,temp4;
unit choice=0;
go {
commence;
post(″Enter value of n : ″);
capture(#n);
phase(n > 50) {
post(″Must not Exceeded to 50!\n″);
post(″Enter value of n : ″);
capture(#n);
}
temp=n/2;
post(″ ″);
inquire(k=temp; k<=n; k++) {
k=k+2;
temp2=n-k;
inquire(j=1; j<temp2; j++) {
j=j+2;
post(″ ″);
}
inquire(j=1;
j<=k; j++) {
post(″*″);
}
inquire(j=1;
j<=temp2; j++)
{
post(″ ″);
}
inquire(j=1;
j<=k; j++) {
post(″*″);
}
post(″\n″);
}
inquire(i=n;
i>=1; i--) {
inquire(j=i;
j<n; j++) {
post(″ ″);
}
temp3=(i*2);
post(″ ″);
inquire(j=1; j<=temp3; j++) {
post(″*″);
}
post(″\n″);
}
go {
company ch;
post(″\n\n\t\t\tTRY AGAIN? [Y] Yes or [N]
No: ″);
capture(#ch);
inorder((ch = = ″Y″) || (ch = = ″y″)) {
i=0;
j=0;
n=0;
k=0;
temp=0;
temp2=0;
temp3=0;
temp4=0;
choice = 1;
}
otherorder((ch = = ″N″) || (ch = = ″n″)) {
choice = 0;
}
order {
post(″\n\t\t\tError Input!″);
choice = 3;
}
} phase(choice
= = 3);
} phase(choice
!= 0);
post(″\n\t\t\tGOODBYE!!″);
} deploy();
```

Print screen:

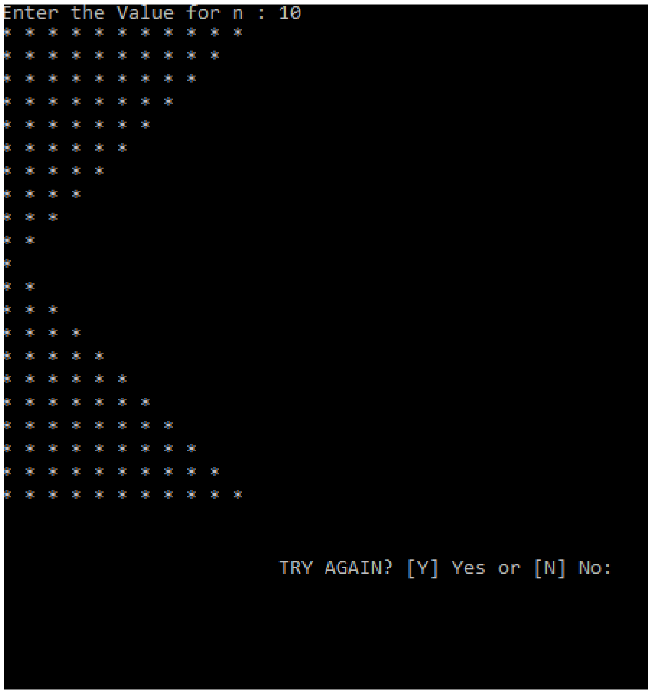


```
Enter value of n : 20
    ************        ************
  ***************     ***************
 ******************  ******************
******************************************
 ****************************************
  **************************************
   ************************************
    **********************************
     ********************************
      ******************************
       ****************************
        **************************
         ************************
          **********************
           ********************
            ******************
             ****************
              **************
               ************
                **********
                 ********
                  ******
                   ****
                    **


                        TRY AGAIN? [Y] Yes or [N] No: _
```

Program # 3:
Source Code:

```
PrimaryMission() {
unit i,j,k,n,temp=0;
unit choice=0;
go {
commence;
post("Enter the Value for n : ");
capture(#n);
temp=temp-n;
inquire(i=temp;i<=n;i++) {
k=i;
inorder(k<0) {
k= k* ~1;
}
inquire(j=0;j<=n;j++) {
inorder(k>=j) {
post("* ");
}
order {
post(" ");
}
}
post("\n");
}

go {
company ch;
post("\n\n\t\t\tTRY AGAIN? [Y] Yes or [N]
No: ");
capture(#ch);
inorder((ch = = "Y") || (ch = = "y")) {
i=0;
j=0;
k=0;
n=0;
temp=0;
choice = 1;
}
otherorder((ch = = "N") || (ch = = "n")) {
choice = 0;
}
order {
post("\n\t\t\tError Input!");
choice = 3;
}
} phase(choice = = 3);

} phase(choice != 0);
post("\n\t\t\tGOODBYE!!");
} deploy();
```

Print screen:



```
Enter the Value for n : 10
* * * * * * * * * * *
* * * * * * * * * *
* * * * * * * * *
* * * * * * * *
* * * * * * *
* * * * * *
* * * * *
* * * *
* * *
* *
*
* *
* * *
* * * *
* * * * *
* * * * * *
* * * * * * *
* * * * * * * *
* * * * * * * * *
* * * * * * * * * *
* * * * * * * * * * *

                  TRY AGAIN? [Y] Yes or [N] No:
```

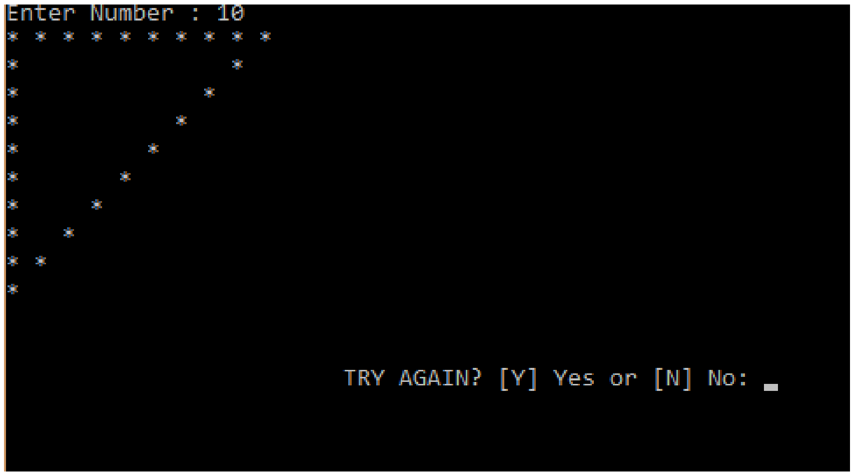
Program # 4:

Source Code:

```
PrimaryMission() {
unit i,j,k,n,temp;
unit choice=0;
go {
commence;
post(″Enter Number : ″);
capture(#n);
inquire(i=1;i<=n;i++) {
inquire(j=1;j<=n;j++) {
temp = n + 1-i;
inorder(j <= temp) {
inorder((i = =1) || (j = = 1) || (j = =
temp)) {
post(″* ″);
}
order {
post(″  ″);
}
}
}
post(″\n″);
}

go {
company ch;
post(″\n\n\t\t\tTRY AGAIN? [Y] Yes or [N]
No: ″);
capture(#ch);
inorder((ch = = ″Y″) || (ch = = ″y″)) {
i=0;
j=0;
k=0;
n=0;
temp=0;
choice = 1;
}
otherorder((ch = = ″N″) || (ch = = ″n″)) {
choice = 0;
}
order {
post(″\n\t\t\tError Input!″);
choice = 3;
}
} phase(choice = = 3);
} phase(choice != 0);
post(″\n\t\t\tGOODBYE!!″);
} deploy();
```

Print Screen:

Program # 5:

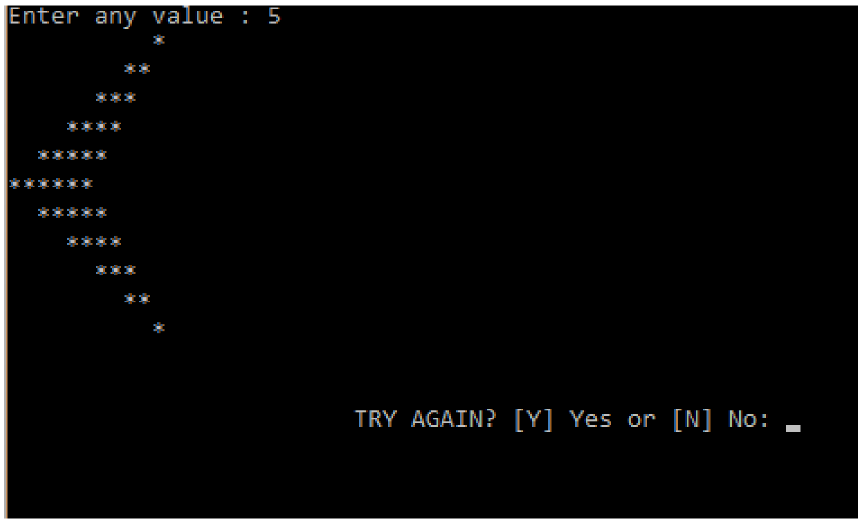


```
Enter Number : 10
* * * * * * * * * *
*               *
*             *
*           *
*         *
*       *
*     *
*   *
* *
*

            TRY AGAIN? [Y] Yes or [N] No:
```

Source Code:

```
PrimaryMission() {
unit i,j,k,num,temp=0;
unit choice=0;
go {
commence;
post("Enter any value : ");
capture(#num);
temp=temp-num;
inquire(i=temp;i<=num;i++) {
k=i;
inorder(k<0) {
k = k * ~1;
}
inquire(j = 0; j <= num; ++j) {
inorder(j<k) {
post("  ");
}
order {
post("*");
}
}
post("\n");
}

go {
company ch;
post("\n\n\t\t\tTRY AGAIN? [Y] Yes or [N]
No: ");
capture(#ch);
inorder((ch = = "Y") || (ch = = "y")) {
i=0;
j=0;
k=0;
num=0;
temp=0;
choice = 1;
}
otherorder((ch = = "N") || (ch = = "n")) {
choice = 0;
}
order {
post("\n\t\t\tError Input!");
choice = 3;
}
} phase(choice = = 3);
} phase(choice != 0);
post("\n\t\t\tGOODBYE!!");
} deploy();
```

Print Screen:



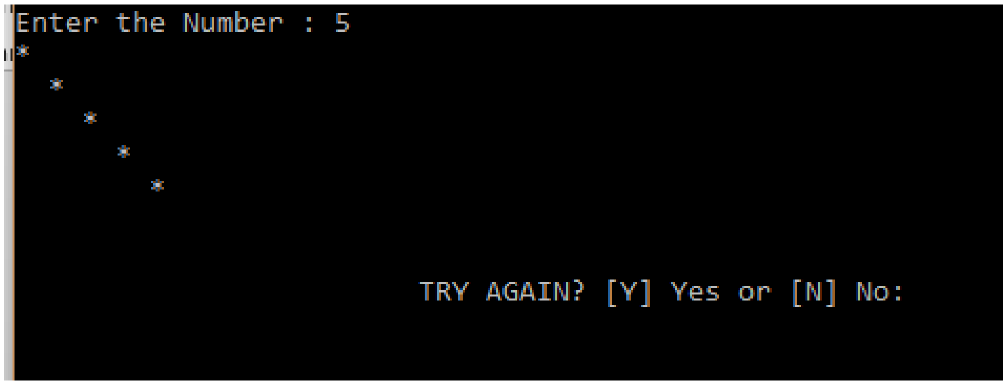
```
Enter any value : 5
          *
        **
      ***
    ****
  *****
******
  *****
    ****
      ***
        **
          *



                        TRY AGAIN? [Y] Yes or [N] No: _
```

Program # 6:
Source Code:

```
PrimaryMission() {
```

```
unit i,j,k,num;
unit choice=0;
go {
commence;
post("Enter the Number : ");
capture(#num);
inquire(i=1;i<=num;i++) {
inquire(j=1;j<=num;j++) {
inorder(i==j) {
post("* ");
}
order {
post("  ");
}
}
post("\n");
}

go {
company ch;
post("\n\n\t\t\tTRY AGAIN? [Y] Yes or [N]
No: ");
capture(#ch);
inorder((ch = = "Y") || (ch = = "y")) {
i=0;
j=0;
k=0;
num=0;
choice = 1;
}
otherorder((ch = = "N") || (ch = = "n")) {
choice = 0;
}
order {
post("\n\t\t\tError Input!");
choice = 3;
}
} phase(choice = = 3);
} phase(choice != 0);
post("\n\t\t\tGOODBYE!!");
} deploy();
```

Print Screen:

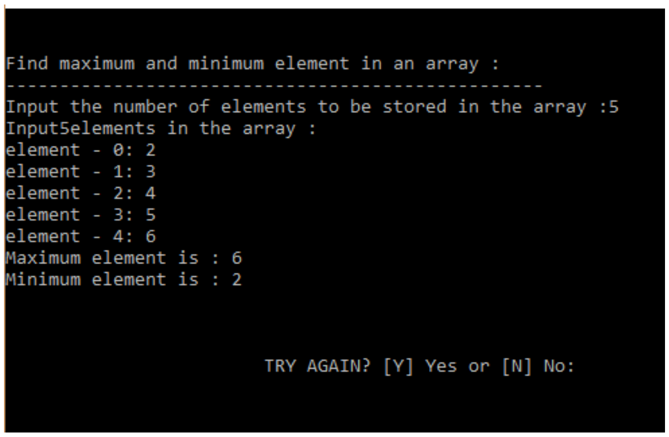


```
Enter the Number : 5
*
  *
    *
      *
        *


                        TRY AGAIN? [Y] Yes or [N] No:
```

Program # 7:
Source Code:

```
unit arr1[100];

PrimaryMission() {
unit i, mx, mn, n;
unit choice=0;
go {
commence;
post(″\n\nFind maximum and minimum
element in an array :\n″);
post(″--------------------------------
-----------------\n″);
post(″Input the number of elements to
be stored in the array :″);
capture(#n);
post(″Input″ + n + ″elements in the
array :\n″);
inquire(i=0;i<n;i++) {
post(″element - ″ + i + ″: ″);
capture(#arr1[i]);
}
mx = arr1[0];
mn = arr1[0];

inquire(i=1; i<n; i++) {
inorder(arr1[i] > mx) {
mx = arr1[i];
}
inorder(arr1[i] < mn) {
mn = arr1[i];
}
}
post(″Maximum element is : ″ + mx +
″\n″);
post(″Minimum element is : ″ + mn +
″\n\n″);

go {
company ch;
post(″\n\n\t\t\tTRY AGAIN? [Y] Yes or
[N] No: ″);
capture(#ch);
inorder((ch = = ″Y″) || (ch = = ″y″)) {
i=0;
mx=0;
mn=0;
n=0;
choice = 1;
}
otherorder((ch = = ″N″) || (ch = =
″n″)) {
choice = 0;
}
order {
post(″\n\t\t\tError Input!″);
choice = 3;
}

} phase(choice = = 3);

} phase(choice != 0);
post(″\n\t\t\tGOODBYE!!″);
} deploy();
```

Print Screen:



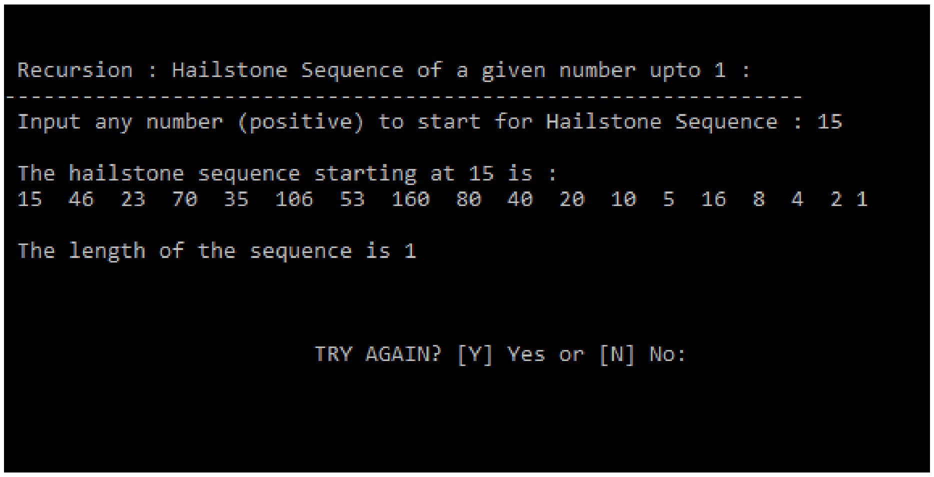
```
Find maximum and minimum element in an array :
------------------------------------------------
Input the number of elements to be stored in the array :5
Input5elements in the array :
element - 0: 2
element - 1: 3
element - 2: 4
element - 3: 5
element - 4: 6
Maximum element is : 6
Minimum element is : 2


                    TRY AGAIN? [Y] Yes or [N] No:
```

Program # 8:
Source Code:

```
unit getNextValue(unit aNum) {
unit i;
i = aNum;
unit temp;
temp = i%2;
inorder(temp = = 0) {
i = i/2;
}
order {
i = 3 * i + 1;
}
backup(i);
}
```

```
miss getHailstone(unit aNum) {
unit hlSe;
hlSe = aNum;
unit temp;
inorder(hlSe = = 1) {
post(hlSe + " ");
}
order {
post(" " + hlSe + " ");
temp = getNextValue(hlSe);
getHailstone(temp);
}
}

unit countLength(unit aNum) {
unit hlSe;
hlSe = aNum;
unit cnt = 0;
unit temp;
inorder(hlSe = = 1) {
cnt = 1;
}
order {
temp = getNextValue(hlSe);
cnt = cnt + countLength(temp);
}
backup(cnt);
}

PrimaryMission() {
unit aNum;
unit temp;
unit choice=0;
go {
commence;
post("\n\n Recursion : Hailstone
Sequence of a given number upto 1 :
\n");
post("--------------------------------
---------------------------- \n");
post(" Input any number (positive) to
start for Hailstone Sequence : ");
capture(#aNum);

phase(aNum <= 0) {
post(" Input any number (*positive) to
start for Hailstone Sequence : ");
capture(#aNum);
}

post("\n The hailstone sequence
starting at " + aNum + " is : \n");
getHailstone(aNum);
post("\n\n");
temp = countLength(aNum);
post(" The length of the sequence is "
+ temp + "\n\n");

go {
company ch;
post("\n\n\t\t\tTRY AGAIN? [Y] Yes or
[N] No: ");
capture(#ch);
inorder((ch = = "Y") || (ch = = "y")) {
aNum=0;
temp=0;
choice = 1;
}
otherorder((ch = = "N") || (ch = =
"n")) {
choice = 0;
}
order {
post("\n\t\t\tError Input!");
choice = 3;
}

} phase(choice = = 3);

} phase(choice != 0);
post("\n\t\t\tGOODBYE!!");
} deploy();
```

Print Screen:



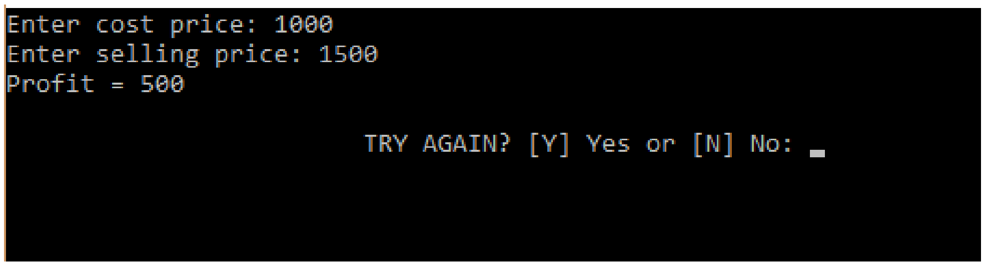
```
Recursion : Hailstone Sequence of a given number upto 1 :
--------------------------------------------------------------
Input any number (positive) to start for Hailstone Sequence : 15

The hailstone sequence starting at 15 is :
15  46  23  70  35  106  53  160  80  40  20  10  5  16  8  4  2 1

The length of the sequence is 1



                      TRY AGAIN? [Y] Yes or [N] No:
```

Program # 9:
Source Code:

```
PrimaryMission() {
unit cp,sp, amt;
unit choice=0;
go {
commence;
post("Enter cost price: ");
capture(#cp);
post("Enter selling price: ");
capture(#sp);
inorder(sp > cp) {
amt = sp - cp;
post("Profit = " + amt);
}
otherorder(cp > sp) {
amt = cp - sp;
post("Loss = " + amt);
}
order {
post("\nNo Profit No Loss.");
}

go {
company ch;
post("\n\n\t\t\tTRY AGAIN? [Y] Yes or
[N] No: ");

capture(#ch);
inorder((ch = = "Y") || (ch = = "y")) {
cp=0;
sp=0;
amt=0;
choice = 1;
}
otherorder((ch = = "N") || (ch = =
"n")) {
choice = 0;
}
order {
post("\n\t\t\tError Input!");
choice = 3;
}

} phase(choice = = 3);

} phase(choice != 0);
post("\n\t\t\tGOODBYE!!");
} deploy();
```

```
post(″\n\n\t\t\tTRY AGAIN? [Y] Yes or
[N] No: ″);
```

Print Screen:



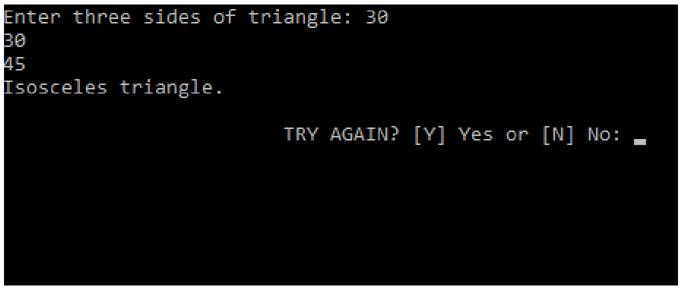
```
Enter cost price: 1000
Enter selling price: 1500
Profit = 500

                    TRY AGAIN? [Y] Yes or [N] No: _
```

Program #10:

Source Code:

```
PrimaryMission() {
unit a, b, c;
unit choice=0;
go {
commence;
post(″Enter three sides of triangle:
″);
capture(#a);
capture(#b);
capture(#c);
inorder((a==b) & (b==c)) {
post(″Equilateral triangle.″);
}
otherorder((a==b) || (a==c) || (b==c))
{
post(″Isosceles triangle.″);
}
order {
post(″Scalene triangle.″);
}
go {
company ch;
capture(#ch);
inorder((ch = = ″Y″) || (ch = = ″y″)) {
a=0;
b=0;
c=0;
choice = 1;
}
otherorder((ch = = ″N″) || (ch = =
″n″)) {
choice = 0;
}
order {
post(″\n\t\t\tError Input!″);
choice = 3;
}

} phase(choice = = 3);

} phase(choice != 0);
post(″\n\t\t\tGOODBYE!!″);
} deploy();
```

Print Screen:



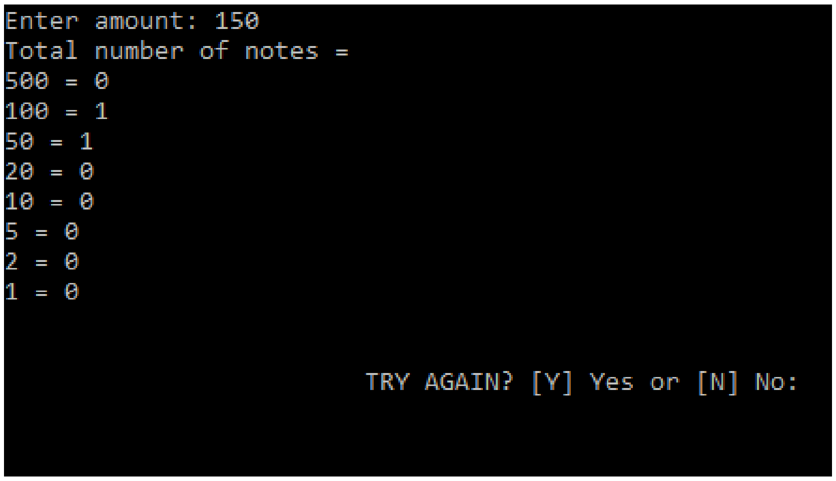
```
Enter three sides of triangle: 30
30
45
Isosceles triangle.

                    TRY AGAIN? [Y] Yes or [N] No:
```

Program # 11:

Source Code:

```
PrimaryMission() {
unit amount;
unit note500=0, note100=0, note50=0,
note20=0, note10=0, note5=0, note2=0,
note1=0;
unit choice=0;
go {
commence;
post("Enter amount: ");
capture(#amount);
inorder(amount >= 500) {
note500 = amount/500;
amount = amount - note500 * 500;
}
inorder(amount >= 100) {
note100 = amount/100;
amount = amount - note100 * 100;
}
inorder(amount >= 50) {
note50 = amount/50;
amount = amount - note50 * 50;
}
inorder(amount >= 20) {
note20 = amount/20;
amount = amount - note20 * 20;
}
inorder(amount >= 10) {
note10 = amount/10;
amount = amount - note10 * 10;
}
inorder(amount >= 5) {
note5 = amount/5;
amount = amount - note5 * 5;
}
inorder(amount >= 2) {
note2 = amount /2;
amount = amount - note2 * 2;
}
inorder(amount >= 1) {
note1 = amount;
}
post("Total number of notes = \n");
post("500 = " + note500 + "\n");
post("100 = " + note100 + "\n");
post("50 = " + note50 + "\n");
post("20 = " + note20 + "\n");
post("10 = " + note10 + "\n");
post("5 = " + note5 + "\n");
post("2 = " + note2 + "\n");
post("1 = " + note1 + "\n");

go {
company ch;
post("\n\n\t\t\tTRY AGAIN? [Y] Yes or
[N] No: ");
capture(#ch);
inorder((ch = = "Y") || (ch = = "y")) {
note500=0;
note100=0;
note50=0;
note20=0;
note10=0;
note5=0;
note2=0;
note1=0;
choice = 1;
}
otherorder((ch = = "N") || (ch = =
"n")) {
choice = 0;
}
order {
post("\n\t\t\tError Input!");
choice = 3;
}
} phase(choice = = 3);
} phase(choice != 0);
post("\n\t\t\tGOODBYE!!");
} deploy();
```

Print Screen:

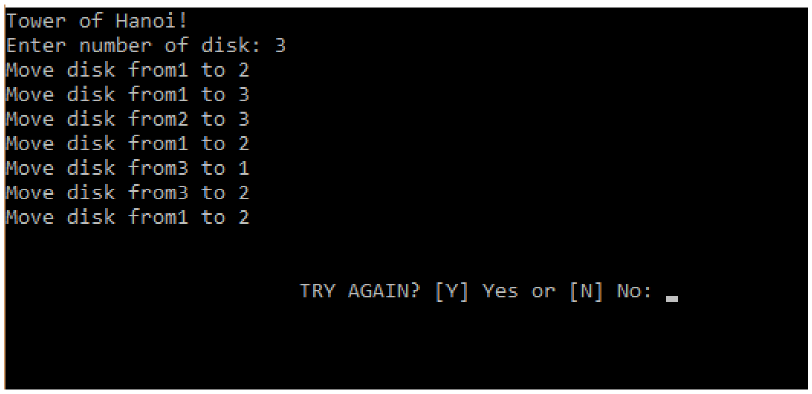


```
Enter amount: 150
Total number of notes =
500 = 0
100 = 1
50 = 1
20 = 0
10 = 0
5 = 0
2 = 0
1 = 0

                        TRY AGAIN? [Y] Yes or [N] No:
```

Program # 12:
Source Code:

```
miss hanoi(unit ndisk, joe source, joe
target, joe other) {
inorder(ndisk > 0) {
ndisk=ndisk-1;
hanoi(ndisk, source, other, target);
post("Move disk from" + source + " to "
+target+ "\n");
hanoi(ndisk, other, target, source);
}
}

PrimaryMission() {
unit ndisk;
unit choice=0;
go {
commence;
post("Tower of Hanoi!\n");
post("Enter number of disk: ");
capture(#ndisk);
hanoi(ndisk, '1', '2', '3');

go {
company ch;
post("\n\n\t\t\tTRY AGAIN? [Y] Yes or
[N] No: ");
capture(#ch);
inorder((ch = = "Y") || (ch = = "y")) {
ndisk=0;
choice = 1;
}
otherorder((ch = = "N") || (ch = =
"n")) {
choice = 0;
}
order {
post("\n\t\t\tError Input!");
choice = 3;
}

} phase(choice = = 3);

} phase(choice != 0);
post("\n\t\t\tGOODBYE!!");
} deploy();
```

Print Screen:



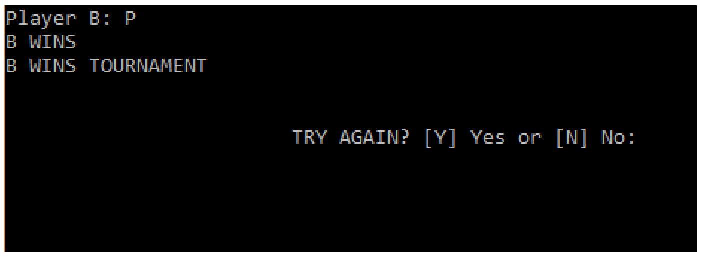
```
Tower of Hanoi!
Enter number of disk: 3
Move disk from1 to 2
Move disk from1 to 3
Move disk from2 to 3
Move disk from1 to 2
Move disk from3 to 1
Move disk from3 to 2
Move disk from1 to 2


                TRY AGAIN? [Y] Yes or [N] No:
```

Program # 13:
Source Code:

```
unit who_wins(company a, company b) {
unit num=0;
inorder((a = = ″P″) & (b = = ″R″)) {
num = 1;
}
inorder((a = = ″P″) & (b = = ″S″)) {
num = 2;
}
inorder((a = = ″R″) & (b = = ″P″)) {
num = 2;
}
inorder((a = = ″R″) & (b = = ″S″)) {
num = 1;
 }
inorder((a = = ″S″) & (b = = ″R″)) {
num = 2;
}
inorder((a = = ″S″) & (b = = ″P″)) {
num = 1;
}
backup(num);
}

PrimaryMission() {
unit temp=0;
unit count_A = 0;
unit count_B = 0;
company pl1;
company pl2;
unit choice=0;
go {
commence;
post(″ROCK [R], PAPER [P] and SCISSOR's
[S] TOURNAMENT!!!\n″);
post(″Player A: ″);
capture(#pl1);
commence;
post(″Player B: ″);
capture(#pl2);
temp = who_wins(pl1,pl2);
inorder(temp = = 1 ) {
post(″A WINS\n″);
count_A++;
}
otherorder(temp = = 2 ) {
```

```
post(″B WINS\n″);
count_B++;
}
order {
post(″DRAW\n″);
}
inorder(count_A > count_B) {
post(″A WINS TOURNAMENT\n″);
}
order {
post(″B WINS TOURNAMENT\n″);
}

go {
company ch;
post(″\n\n\t\t\tTRY AGAIN? [Y] Yes or
[N] No: ″);
capture(#ch);
inorder((ch = = ″Y″) || (ch = = ″y″)) {
temp=0;
count_A = 0;
count_B = 0;
pl1 = ″ ″;
pl2 = ″ ″;
choice = 1;
}
otherorder((ch = = ″N″) || (ch = =
″n″)) {
choice = 0;
}
order {
post(″\n\t\t\tError Input!″);
choice = 3;
}

} phase(choice = = 3);

} phase(choice != 0);
post(″\n\t\t\tGOODBYE!!″);
} deploy();
```

Print Screen:

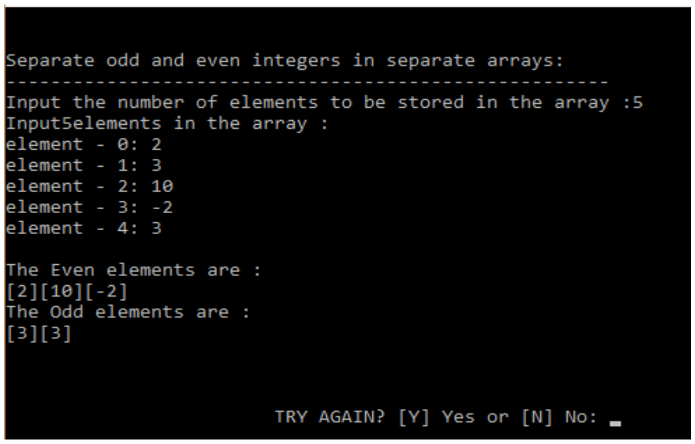


```
Player B: P
B WINS
B WINS TOURNAMENT


                        TRY AGAIN? [Y] Yes or [N] No:
```

Program # 14:
Source Code:

```
unit arr1[10];
unit arr2[10];
unit arr3[10];

PrimaryMission() {
unit i,j=0,k=0,n,temp;
unit choice=0;
go {
commence;
```

```
post("\n\nSeparate odd and even
integers in separate arrays:\n");
post("--------------------------------
---------------------\n");
post("Input the number of elements to
be stored in the array :");
capture(#n);
post("Input" + n + "elements in the
array :\n");
inquire(i=0;i<n;i++) {
post("element - " + i + ": ");
capture(#arr1[i]);
}
inquire(i=0;i<n;i++) {
temp = arr1[i] % 2;
inorder(temp = = 0) {
arr2[j] = arr1[i];
j++;
}
order {
arr3[k] = arr1[i];
k++;
}
}
post("\nThe Even elements are : \n");
inquire(i=0;i<j;i++) {
post("[" + arr2[i] + "]");
}
```



```
Separate odd and even integers in separate arrays:
----------------------------------------------------
Input the number of elements to be stored in the array :5
Input5elements in the array :
element - 0: 2
element - 1: 3
element - 2: 10
element - 3: -2
element - 4: 3

The Even elements are :
[2][10][-2]
The Odd elements are :
[3][3]


                        TRY AGAIN? [Y] Yes or [N] No:
```

```
post("\nThe Odd elements are :\n");
inquire(i=0;i<k;i++) {
post("[" + arr3[i] + "]");
}
post("\n\n");
```

```
go {
company ch;
post("\n\n\t\t\tTRY AGAIN? [Y] Yes or
[N] No: ");
capture(#ch);
inorder((ch = = "Y") || (ch = = "y")) {
i=0;
j=0;
k=0;
n=0;
temp=0;
choice = 1;
}
otherorder((ch = = "N") || (ch = =
"n")) {
choice = 0;
}
order {
post("\n\t\t\tError Input!");
choice = 3;
}

} phase(choice = = 3);

} phase(choice != 0);
post("\n\t\t\tGOODBYE!!");
} deploy();
```

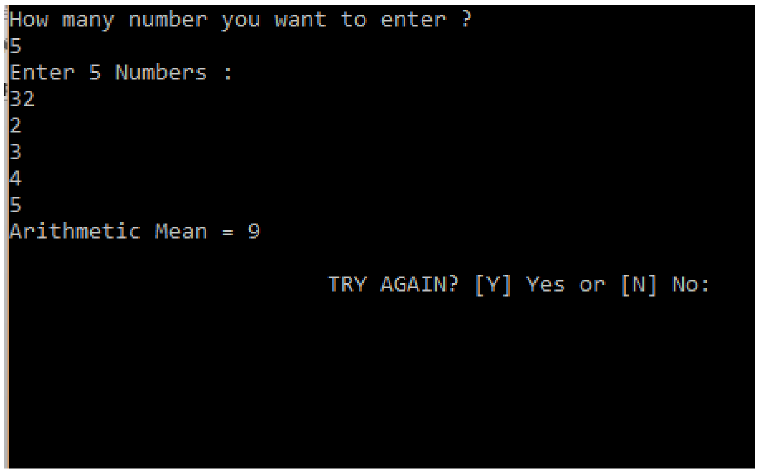
Print Screen:

Program # 15:
Source Code:

```
unit arr[50];
PrimaryMission() {
```

```
unit n, i, sum=0;
unit choice;
go {
commence;
post("How many number you want to enter
?\n");
capture(#n);
post("Enter " + n + " Numbers :\n");
inquire(i=0; i<n; i++) {
capture(#arr[i]);
sum=sum+arr[i];
}
unit armean;
armean = sum/n;
post("Arithmetic Mean = " + armean);

go {
company ch;
post("\n\n\t\t\tTRY AGAIN? [Y] Yes or
[N] No: ");
capture(#ch);
inorder((ch = = "Y") || (ch = = "y")) {
n=0;
i=0;
sum=0;
choice = 1;
}
otherorder((ch = = "N") || (ch = =
"n")) {
choice = 0;
}
order {
post("\n\t\t\tError Input!");
choice = 3;
}

} phase(choice = = 3);

} phase(choice != 0);
post("\n\t\t\tGOODBYE!!");
} deploy();
```

Print Screen:

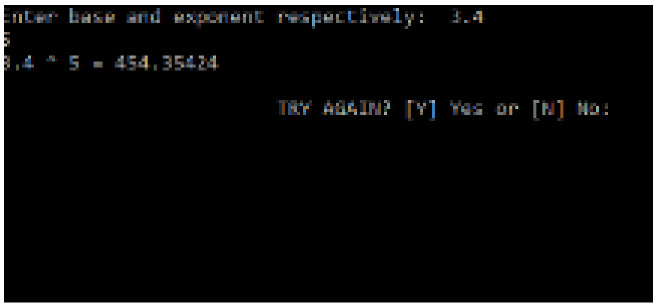


```
How many number you want to enter ?
5
Enter 5 Numbers :
32
2
3
4
5
Arithmetic Mean = 9

                        TRY AGAIN? [Y] Yes or [N] No:
```

Program # 16:
Source Code:

```
PrimaryMission() {
unit exponent;
digit base1, result = 1.0;
unit choice;
go {
commence;
post("Enter base and exponent
respectively:  ");
capture(#base1);
capture(#exponent);
post(base1 + " ^ " + exponent + " = ");
phase(exponent != 0) {
result = result * base1;
exponent--;
}
post(result);

go {
company ch;
post("\n\n\t\t\tTRY AGAIN? [Y] Yes or
[N] No: ");
capture(#ch);
inorder((ch = = ″Y″) || (ch = = ″y″)) {
exponent=0;
base1=0;
result = 1.0;
choice = 1;
}
otherorder((ch = = ″N″) || (ch = =
″n″)) {
choice = 0;
}
order {
post(″\n\t\t\tError Input!″);
choice = 3;
}

} phase(choice = = 3);

} phase(choice != 0);
post(″\n\t\t\tGOODBYE!!″);
} deploy();
```

Print Screen:



```
nter base and exponent respectively:  3.4
5
.4 ^ 5 = 454.35424

                         TRY AGAIN? [Y] Yes or [N] No:
```