Pascal Triangle

```
import java.util.Scanner;
public class P1
{
public static void main(String args[])
{
Scanner sc=new Scanner(System.in);
System.out.print("Enter No : " );
int no=sc.nextInt();
 int c=1;
for(int i=0;i<no;i++)
{
  for(int j=0;j<=i;j++)
  {
   if(j==0 || i==0)
      c=1;
  else
    c=c*(i-j+1)/j;
 System.out.print(c+"  ");
   } // j++
 System.out.println();
  } // i++
}}
```

output:

```
Enter No : 5
     1
    1  1
```

```
   1  2  1
  1  3  3  1
 1  4  6  4  1
```

Ambiguity Error

```
class A
{
void meth()
{
}
}
class B
{
void meth()
{
}
}
class C:A,B
{
void meth()
{
meth(); // ambiguity error - Interface
}
}
```

Collection

> similar to array

> but dynamic

> can store more than one element in a single variable name in different datatype and different mode.

> collections are stored in java.util package

```
import java.util.ArrayList;
import java.util.List;
public class P1
{
public static void main(String args[])
{
List<String> lst =new ArrayList<>();

lst.add("Rithikaa");
lst.add("Varshinii");
lst.add("100");

 System.out.println(lst);
}}
```

```
import java.util.*;
class P2 {
    public static void main(String[] args) {
        P2 obj = new P2();
        List<List<Integer>> lst = obj.meth(15);
        for (List<Integer> row : lst) {
            System.out.println(row);
        }
    }
 public List<List<Integer>> meth(int no) {
        List<List<Integer>> result = new ArrayList<>();
        for (int i = 1; i <= no; i++) {
            List<Integer> row = new ArrayList<>();
            for (int j = 1; j <= i; j++) {
                row.add(j);
            }
            result.add(row);
        } // i++
        return result;
    }
}
```

Execise : Pascal Triangle - Leetcode - 118

Pascal Triangle II - Leetcode - 119

```
class Solution {
    public List<Integer> getRow(int rowIndex) {
        List<Integer> lst = new ArrayList<>();
        int first = 1;
        lst.add(first);
        long res = 1;
        for(int col=1; col<=rowIndex; col++){
            res = res * (rowIndex - col + 1)/col;
            lst.add((int)res);
        }
        return lst;
    }
}
```

Random in c:

```
#include <stdio.h>
#include<stdlib.h>
int main()
{
   for(int i=1;i<=5;i++)
   {
     printf("\n%d", rand()%100);
   }
   return 0;
}
```

Random in Python

```
import random

for i in range(5):
   print(random.randint(1,50))
```

Random in java

```
import java.util.Random;

class Test1 {
public static void main(String[] args)
{
    Random rand = new Random();
   for(int i=1;i<=5;i++)
   {
System.out.println(rand.nextInt(1,50));
  }
System.out.println("Four Digit OTP : " + rand.nextInt(1000,10000));


}
}
```

Write a program in java with following rules:

> first asks the user to enter a username.

> Then allow the user to guess a number between 1 and 10.

> The user is allowed to enter a number only 3 times.

> For each attempt, the program should generate a random number between 1 and 10.

> If the user’s number matches the random number, print “You win”.

> If the numbers do not match, print “Better luck next time”.

> After 3 attempts, regardless of the result, print “Game Over” and terminate the program.

Life Time Calculation:

```
import java.time.*;
import java.time.temporal.ChronoUnit;
import java.time.format.DateTimeFormatter;
import java.util.Locale;
import java.util.Scanner;

class LifeGame
{
public static void main(String args[])
{
```

```
Scanner sc=new Scanner(System.in);

LocalDateTime now=LocalDateTime.now();

LocalDate today=now.toLocalDate();

System.out.println("Now : " + now);

System.out.println("Today : " + today);

System.out.println("Enter DOB (dd-MM-yyyy) : ");

String dobstr=sc.next();

LocalDate dob=LocalDate.parse(dobstr,DateTimeFormatter.ofPattern("dd-MM-yyyy"));

LocalDateTime sd=dob.atStartOfDay();

System.out.println("DOB : " + dobstr);

System.out.println("DOB : " + dob);

System.out.println("Start Time : " + sd);

Period p=Period.between(dob,today);

System.out.println("Age : " + p.getYears());

System.out.println("Born Day : " + dob.getDayOfWeek());

long daysLived = ChronoUnit.DAYS.between(dob,today);

System.out.println("Days Lived : " + daysLived);

System.out.println("Months lived : "+ ChronoUnit.MONTHS.between(dob, today));

System.out.println("Exact age as at : " + p.getYears()+" years "+ p.getMonths()+" months " + p.getDays()
+ " days ");

Duration du  = Duration.between(sd, now);

long totSec = du.getSeconds();

long hours = totSec/ 3600;

long minutes = (totSec % 3600) / 60;

long seconds = totSec % 60;

System.out.println("Time alive : "+ hours + "  hours,"+ minutes+"  minutes,"+ seconds +"  secs ");


double sleepFactor = 0.33;
```

```
long sl = Math.round(daysLived * sleepFactor);
long hsl = sl * 24;
double ysl = sl / 365.25;
double msl = ysl * 12;


System.out.println("Approx hours slept : " + hsl);
System.out.println("Days sleeping      : " + sl);
System.out.println("Years asleep       : " + String.format("%.2f", ysl));
System.out.println("Months asleep      : " + String.format("%.2f", msl));
 // Heart Beats (Avg 72 bpm)
System.out.println("Approx number of heart beats : "+ totSec * 72 / 60);
int rage = 60;
System.out.println("Retirement Age"+ rage);
LocalDate rdate = dob.plusYears(rage).minusDays(1);
long ryear = ChronoUnit.YEARS.between(today, rdate);
if (rdate.isBefore(today))
      ryear = 0;
long rdays = ChronoUnit.DAYS.between(today, rdate);
    if (rdays < 0)
      rdays = 0;
System.out.println("Years to Retirement : "+ ryear);
System.out.println( "Days to Retirement : "+ rdays);
System.out.println("Retirement Date : "+rdate.format(DateTimeFormatter.ofPattern("dd-MM-yyyy")));
long workDays = (long) (rdays * 5.0 / 7.0);
System.out.println("Approx number of workdays : "+ workDays);


}
}
```
