# Activity 1

public static void main(String[] args) {

// create Scanner to obtain input from command line

Scanner input = new Scanner( System.in );

int number1=0, number2=0;

System.out.print( "Enter first integer: " );

// prompt

number1 = input.nextInt(); // read first number from user

System.out.print( "Enter second integer: " );

// prompt

number2 = input.nextInt(); // read second number from user

if ( number1 == number2 ) System.out.printf( "%d == %d\n", number1, number2 );

if ( number1 != number2 ) System.out.printf( "%d != %d\n", number1, number2 );

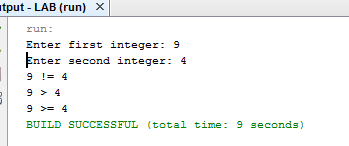
if ( number1 < number2 ) System.out.printf( "%d < %d\n", number1, number2 );

if ( number1 > number2 ) System.out.printf( "%d > %d\n", number1, number2 );

if ( number1 <= number2 ) System.out.printf( "%d <= %d\n", number1, number2 );

if ( number1 >= number2 ) System.out.printf( "%d >= %d\n", number1, number2 );

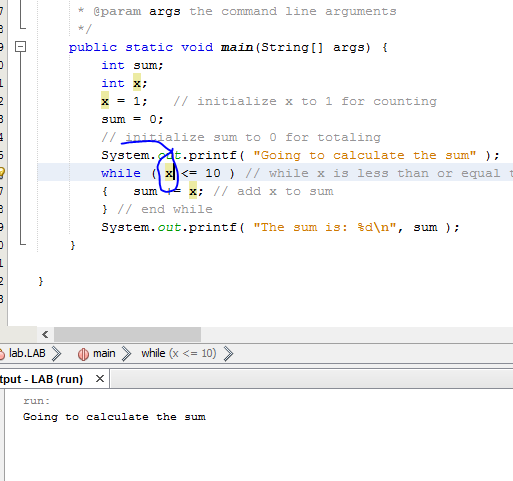
}



# Activity 2

The error is logical one since the while loop run infinite many times

Correct: while(sum<=10)



public static void main(String[] args) {

int sum;

int x;

x = 1; // initialize x to 1 for counting

sum = 0;

// initialize sum to 0 for totaling

System.out.printf( "Going to calculate the sum" );

while ( x <= 10 ) // while x is less than or equal to 10

{ sum += x; // add x to sum

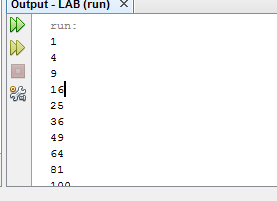
} // end while

System.out.printf( "The sum is: %d\n", sum );

}

# Activity 3

It would print out squares of integers from 1 to 10



# Activity 4

## Output:

run:

Original deck:

2 of Clubs

3 of Clubs

4 of Clubs

5 of Clubs

6 of Clubs

7 of Clubs

8 of Clubs

9 of Clubs

10 of Clubs

Jack of Clubs

Queen of Clubs

King of Clubs

Ace of Clubs

2 of Diamonds

3 of Diamonds

4 of Diamonds

5 of Diamonds

6 of Diamonds

7 of Diamonds

8 of Diamonds

9 of Diamonds

10 of Diamonds

Jack of Diamonds

Queen of Diamonds

King of Diamonds

Ace of Diamonds

2 of Hearts

3 of Hearts

4 of Hearts

5 of Hearts

6 of Hearts

7 of Hearts

8 of Hearts

9 of Hearts

10 of Hearts

Jack of Hearts

Queen of Hearts

King of Hearts

Ace of Hearts

2 of Spades

3 of Spades

4 of Spades

5 of Spades

6 of Spades

7 of Spades

8 of Spades

9 of Spades

10 of Spades

Jack of Spades

Queen of Spades

King of Spades

Ace of Spades

Modified deck:

Jack of Clubs

4 of Hearts

Ace of Hearts

6 of Hearts

8 of Clubs

King of Hearts

6 of Diamonds

Ace of Clubs

7 of Diamonds

Jack of Hearts

5 of Clubs

Ace of Spades

3 of Hearts

5 of Spades

2 of Hearts

9 of Diamonds

9 of Clubs

King of Diamonds

7 of Spades

4 of Diamonds

2 of Spades

2 of Diamonds

7 of Hearts

4 of Spades

Queen of Clubs

8 of Hearts

10 of Diamonds

4 of Clubs

3 of Diamonds

5 of Hearts

8 of Spades

3 of Clubs

Jack of Spades

10 of Spades

6 of Spades

7 of Clubs

10 of Hearts

Jack of Diamonds

King of Spades

Ace of Diamonds

King of Clubs

9 of Hearts

2 of Clubs

3 of Spades

5 of Diamonds

Queen of Hearts

6 of Clubs

Queen of Spades

Queen of Diamonds

9 of Spades

10 of Clubs

8 of Diamonds

public class LAB {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

String [] suit = { "Clubs", "Diamonds", "Hearts", "Spades"};

String [] rank = { "2", "3", "4", "5", "6", "7", "8", "9", "10", "Jack", "Queen", "King", "Ace" };

String [] deck = new String [suit.length \* rank.length];

for (int i = 0; i < suit.length; i++)

for (int j = 0; j < rank.length; j++)

deck [rank.length \* i + j] = rank [j] + " of " + suit[i];

System.out.println("Original deck:");

for (int k = 0; k < (suit.length \* rank.length); k++)

System.out.println(deck[k]);

System.out.println("Modified deck:");

modify\_deck(deck);

for (int k = 0; k < (suit.length \* rank.length); k++)

System.out.println(deck[k]);

}

private static void modify\_deck(String [] deck)

{

int N = deck.length;

for (int i = 0; i < N; i++)

{

int r = i + (int) (Math.random() \* (N - i));

String t = deck[i];

deck[i] = deck[r];

deck[r] = t;

}

}

}

# Task 1

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

int choice=0;

do{

System.out.println("Enter your weight: ");

float weight = scanner.nextFloat();

System.out.println("Enter your height: ");

float height = scanner.nextFloat();

float bmi=weight/(height\*height);

System.out.println("The bmi is: "+ bmi);

if(bmi<18.5){

System.out.println("YOU ARE UNDERWIEGHT");

}

else if(bmi>=18.5 && bmi<=24.9){

System.out.println("YOU ARE NORMAL WEIGHT");

}

else if(bmi>24.9 && bmi <=29.9){

System.out.println("YOU ARE OVERWIEGHT");

}

else if(bmi>29.9){

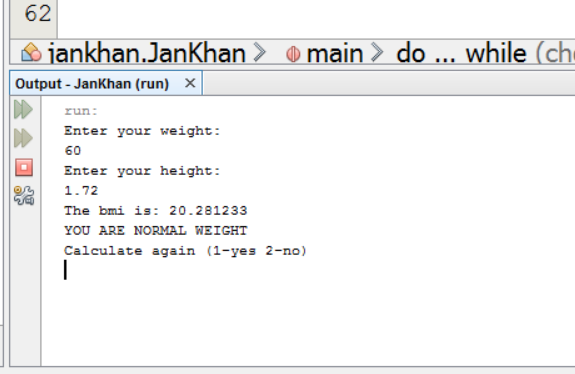
System.out.println("YOU ARE OBESE");

}

System.out.println("Calculate again (1-yes 2-no)");

choice = scanner.nextInt();

}while(choice==1);

}

# Task 2

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

int choice=0;

float totalDistance=0,totalLiters=0;

do{

System.out.println("Enter the distance covered: ");

float distance = scanner.nextFloat();

System.out.println("Enter the number of liters consumed: ");

float liters = scanner.nextFloat();

totalDistance+=distance;

totalLiters+=liters;

System.out.println("The total distance covered:"+ distance);

System.out.println("The total liters used:"+ distance);

float milage=totalDistance/totalLiters;

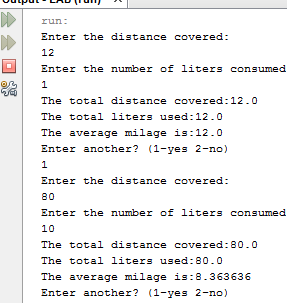
System.out.println("The average milage is:"+ milage);

System.out.println("Enter another? (1-yes 2-no)");

choice = scanner.nextInt();

}while(choice==1);

}



# Task 3

public class LAB {

/\*\*

\* @param args the command line arguments

\*/

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

String [] suit = { "Clubs", "Diamonds", "Hearts", "Spades"};

String [] rank = { "2", "3", "4", "5", "6", "7", "8", "9", "10", "Jack", "Queen", "King", "Ace" };

String [] deck = new String [suit.length \* rank.length];

for (int i = 0; i < suit.length; i++)

for (int j = 0; j < rank.length; j++)

deck [rank.length \* i + j] = rank [j] + " of " + suit[i];

Scanner scanner= new Scanner(System.in);

System.out.println("Enter N: ");

int count=scanner.nextInt();

for(int i=1; i<=count;i++){

System.out.println("Poker:"+i);

modify\_deck(deck);

System.out.println(deck[0]+" , "+deck[1]+" , "+deck[2]+" , "+deck[3]+" , "+deck[4]);

}

}

private static void modify\_deck(String [] deck)

{

int N = deck.length;

for (int i = 0; i < N; i++)

{

int r = i + (int) (Math.random() \* (N - i));

String t = deck[i];

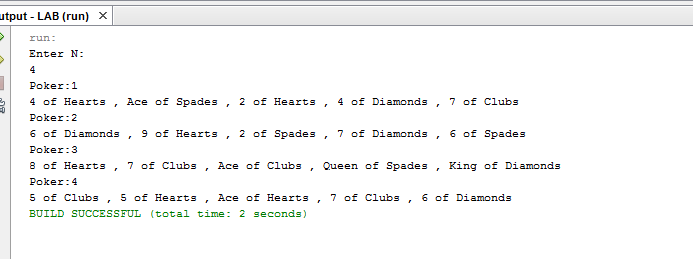
deck[i] = deck[r];

deck[r] = t;

}

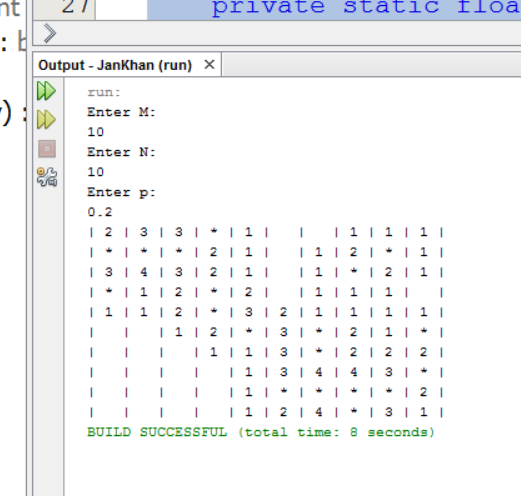
}

}



# 

# Task 4



/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package jankhan;

import java.util.Random;

import java.util.Scanner;

/\*\*

\*

\* @author arifu

\*/

public class JanKhan {

/\*\*

\* @param args the command line arguments

\*/

/\*\*

\* @param args the command line arguments

\*/

//global declarations....

private static boolean[][] grid;

private static int M,N;

private static float p;

private static Scanner scanner = new Scanner(System.in);

public static void main(String[] args) {

System.out.println("Enter M:");

M= scanner.nextInt();

System.out.println("Enter N:");

N= scanner.nextInt();

System.out.println("Enter p:");

p= scanner.nextFloat();

grid= new boolean[M][N];

float totalMines = M\*N\*p;

//populate grid with the number of total mines

for(int i=0; i<totalMines;i++){

int x=getRandomIndex(M);

int y=getRandomIndex(N);

if(grid[x][y]==true){ continue;}

grid[x][y]=true;

}

///printing

for(int i=0; i<M;i++){

System.out.printf("|");

for(int j=0; j<N;j++){

if(get\_cell\_state(i,j)==true){

System.out.printf(" \* |");

}

else{

int neighbors=num\_neighbors(i,j);

if(neighbors==0){

System.out.printf(" |");

}

else{

System.out.printf(" %d |", neighbors);

}

}

}

System.out.printf("\n");

}

}

//modified some functions of LAB 0

public static int getRandomIndex(int x){

Random rand = new Random();

int n = rand.nextInt(x);

return n;

}

public static boolean get\_cell\_state(int x, int y) {

if (x < 0 || x >= M || y < 0 || y >= N)

return false;

return grid[x][y];

}

public static int num\_neighbors(int x, int y) {

int count = 0;

for (int i = x - 1; i <= x + 1; i++){

for (int j = y-1; j <= y+1; j++)

{

if (!(x < 0 || x >= M || y < 0 || y >= N)){

if (!(i == x && j == y)){

if (get\_cell\_state(i, j) == true){

count++;

}

}

}

}

}

return count;

/\* Hint. Use get\_cell\_state(x,y) \*/

}

}