**Experiment 1.1-** WAP to display “Hello, World”

**public class Exp1A {**

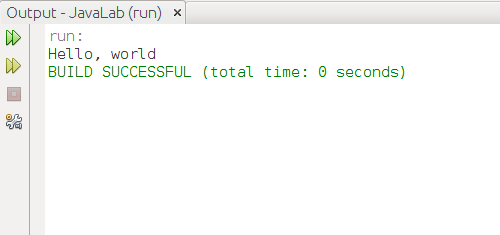
**public static void main(String[] args) {**

**System.out.println("Hello, world");**

**}**

**}**

**OUTPUT**

****

**Experiment 1.2** – WAP to evaluate the following expressions:

1. (a/c)z/a

2. c++ + b/x -y

package javalab;

public class Exp1B {

public static void main(String[] args) {

int a=6,b=2,c=2,x=6,y=4,z=12;

int sol;

sol=(a/c)\*z/a;

System.out.println("(a/c)z/a :"+sol);

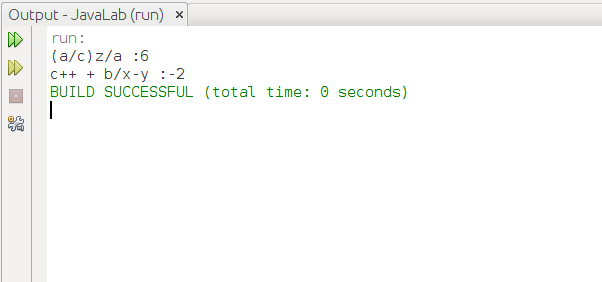
sol=c++ + b/x-y;

System.out.println("c++ + b/x-y :"+sol);

}

}

**OUTPUT**

****

**Experiment 1.3**-WAP to display the values by taking input from console using Scanner class.

**import java.util.Scanner;**

**public class Exp1C {**

**public static void main(String[] args) {**

**Scanner input=new Scanner(System.in);**

**System.out.println("Enter name:");**

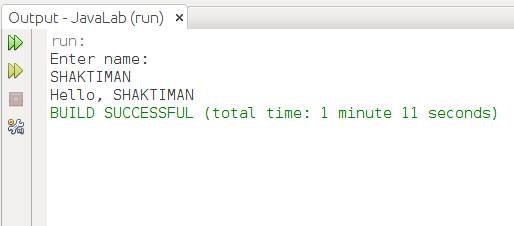
**String name=input.next();**

**System.out.println("Hello, "+name);**

**}**

**}**

**OUTPUT**

****

**Experiment 1.4-** WAP to display “diamond” pattern

public class Exp1D {

public static void main(String[] args) {

for (int i = 0; i < 4; i++) {

for (int j = 4; j > i; j--) {

System.out.print(" ");

}

for (int k = 0; k < (i \* 2 + 1); k++) {

System.out.print("\*");

}

System.out.println();

}

for (int i = 0; i < 9; i++) {

System.out.print("\*");

}

System.out.println();

for (int i = 4; i > 0; i--) {

for (int j = 4; j >= i; j--) {

System.out.print(" ");

}

for (int k = (2 \* (i - 1) + 1); k > 0; k--) {

System.out.print("\*");

}

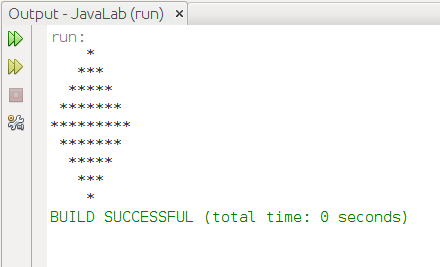
System.out.println();

}

}

}

**OUTPUT**

****

**Experiment 2.1-**Find the largest and smallest element in an araay after getting values from the console.

import java.util.Scanner;

public class Exp2A {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

int size = input.nextInt();

int arr[] = new int[size];

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

System.out.print("arr[" + i + "]:");

arr[i] = input.nextInt();

}

int large = largest(arr, size);

int small = smallest(arr, size);

System.out.println("Largest:" + large + "\nSmallest:" + small);

}

private static int largest(int[] arr, int size) {

int temp = arr[0];

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

if (temp < arr[i]) {

temp = arr[i];

}

}

return temp;

}

private static int smallest(int[] arr, int size) {

int temp = arr[0];

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

if (temp > arr[i]) {

temp = arr[i];

}

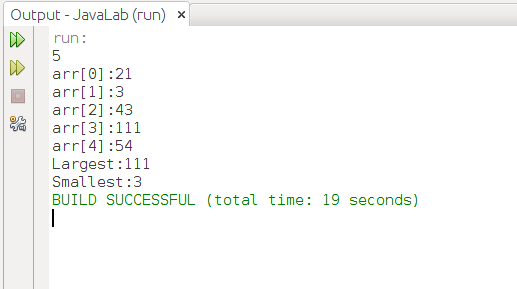
}

return temp;

}

}

**OUTPUT**

****

**Experiment 2.2 –** Write a menu driven program to implement String and StringBuffer operations.

public class Exp2B {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.print("Enter String:");

String str = input.nextLine();

do {

initMenu();

System.out.println("Enter the choice:");

int choice = input.nextInt();

switch (choice) {

case 1:

System.out.println("Length:" + str.length());

break;

case 2:

str = str.toLowerCase();

System.out.println("ToLowerCase:" + str);

break;

case 3:

str = str.toUpperCase();

System.out.println("ToUpperCase:" + str);

break;

case 4:

System.out.print("Enter String:");

String two = input.nextLine();

str = str + two;

//using stringbuilder

System.out.println("Concatinated string:" + str);

break;

case 5:

str = str.trim();

System.out.println("Trim:" + str);

break;

case 6:

System.out.println("Enter beg and end index");

int beg = input.nextInt();

int end = input.nextInt();

System.out.println("Substring(" + beg + "," + end + "):" + str.substring(beg, end));

break;

case 7:

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

int index = input.nextInt();

System.out.println("char At:" + str.charAt(index));

break;

case 8:

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

case 9:

System.exit(0);

default:

System.out.println("wrong choice!!\nTry Again!");

}

} while (true);

}

private static void initMenu() {

System.out.println("1.Length");

System.out.println("2.ToLowerCase");

System.out.println("3.ToUpperCase");

System.out.println("4.Concatination");

System.out.println("5.Trim");

System.out.println("6.SubString");

System.out.println("7.CharAt");

System.out.println("8.Change String");

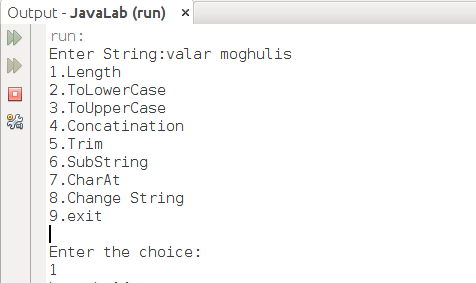
System.out.println("9.exit");

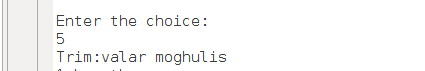
System.out.println();

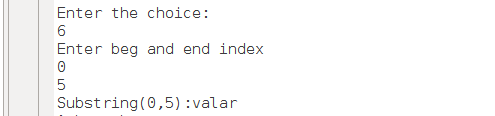
}

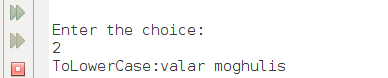
}

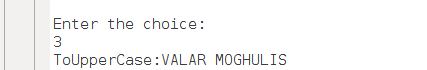
**OUTPUT**

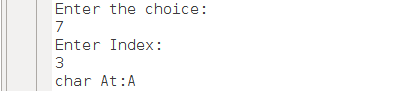












**Experiment 2.3-** Imlement Stack and Queue operations using ArrayList and Iterator class

**STACK**

import java.util.ArrayList;

import java.util.Scanner;

import java.util.Iterator;

import java.util.ListIterator;

public class Exp2C {

public static void main(String[] args) {

Scanner input=new Scanner(System.in);

Stack stack=new Stack();

do{

System.out.println("1.push");

System.out.println("2.pop");

System.out.println("3.exit");

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

int choice=input.nextInt();

switch(choice){

case 1: System.out.println("Enter value:");

int value=input.nextInt();

stack.push(value);

break;

case 2:int pop=stack.pop();

System.out.println("Popped element:"+pop);

break;

case 3:System.exit(0);

default: System.out.println("wrong choice!!");

}

}while(true);

}

}

public class Stack {

private ArrayList<Integer> stack=new ArrayList<>();

private int top=-1;

private static final int SIZE=10;

public void push(int value){

if(top==SIZE){

System.out.println("OVERFLOW --->>");

}else{

top++;

stack.add(value);

System.out.println("PUSHED --->>");

show();

}

}

public int pop() {

int value=-1;

if(top==-1){

System.out.println("UNDERFLOW --->>");

}else{

value=stack.get(top);

stack.remove(top);

top--;

System.out.println("POPPED--->>");

show();

}

return value;

}

public void show(){

ListIterator itr=stack.listIterator(stack.size());

while(itr.hasPrevious()){

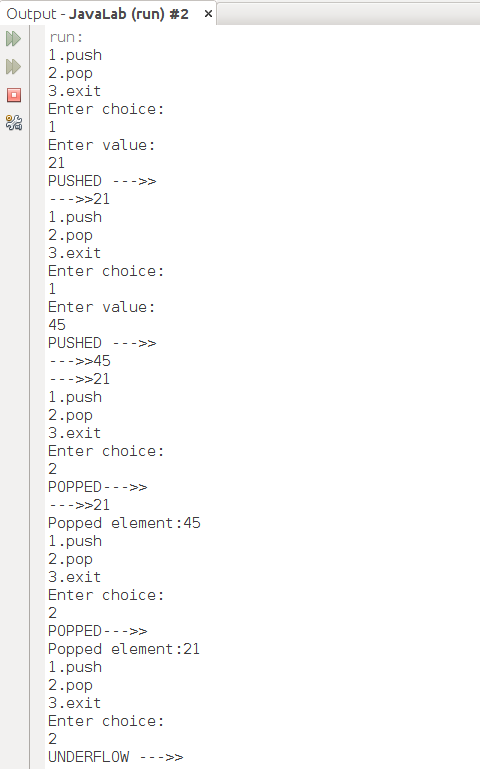
System.out.println("--->>"+itr.previous());

}

}

}

**OUTPUT**



**QUEUE**

import java.util.Scanner;

import java.util.ArrayList;

import java.util.Iterator;

import java.util.ListIterator;

public class Exp2CB {

public static void main(String[] args) {

Scanner input=new Scanner(System.in);

Queue queue=new Queue();

do{

System.out.println("1.Insert");

System.out.println("2.Remove");

System.out.println("3.exit");

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

int choice=input.nextInt();

switch(choice){

case 1: System.out.println("Enter value:");

int value=input.nextInt();

queue.insert(value);

break;

case 2:int pop=queue.remove();

if(pop!=-1)

System.out.println("Popped element:"+pop);

break;

case 3:System.exit(0);

default: System.out.println("wrong choice!!");

}

}while(true);

}

}

public class Queue {

private static final int SIZE = 10;

private ArrayList<Integer> queue = new ArrayList<>(SIZE);

private int rear = -1, front = -1;

public void insert(int value) {

if ((rear + 1) == SIZE) {

System.out.println("OVERFLOW --->>");

} else if (rear == -1) {

front = 0;

rear = 0;

queue.add(value);

System.out.println("INSERTED --->>");

show();

} else {

rear++;

queue.add(value);

System.out.println("INSERTED --->>");

show();

}

}

public int remove() {

int value = -1;

if (front == -1) {

System.out.println("UNDERFLOW --->>");

} else if (front == rear) {

front = -1;

rear = -1;

queue.remove(0);

System.out.println("QUEUE IS EMPTY!!");

} else {

value = queue.get(0);

queue.remove(0);

front++;

System.out.println("REMOVED--->>");

show();

}

return value;

}

public void show() {

Iterator itr = queue.iterator();

while (itr.hasNext()) {

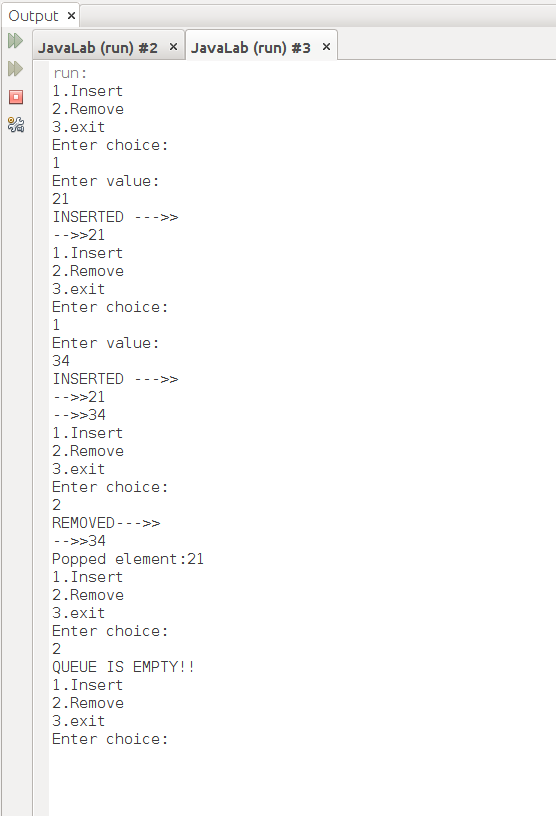
System.out.println("-->>" + itr.next());

}

}

}

**OUTPUT**



**Experiment 3-** (a) Calculate Volume of a box using classes.

(b)-Modify Part (a) by adding constructors and “this” reference.

(c)- Add the member function to the existing class to show the demo of passing objects as a value and returning objects from class.

(d)- Add member function to give a demo of varargs (variable arguments).

import java.util.Scanner;

public class Exp3A {

public static void main(String[] args) {

Scanner input=new Scanner(System.in);

Box box1=new Box();

System.out.println("Enter dimensions of Box1");

System.out.print("Length:");

box1.setLength(input.nextFloat());

System.out.print("Breadth:");

box1.setBreadth(input.nextFloat());

System.out.print("Height:");

box1.setHeight(input.nextFloat());

System.out.println("Volume:"+box1.volume());

System.out.println("Enter dimensions of Box2");

System.out.print("Length:");

float l=input.nextFloat();

System.out.print("Breadth:");

float b=input.nextFloat();

System.out.print("Height:");

float h=input.nextFloat();

Box box2=new Box(l,b,h);

System.out.println("Adding box 1 to Box 2");

Box box3=box1.addBox(box2);

System.out.println(box3.toString());

System.out.println("Display all boxes");

Box.displayBoxes(box1,box2);

}

}

class Box{

float length,breadth,height;

public Box() {

}

//using constructor with this reference PART (B)

public Box(float length, float breadth, float height) {

this.length = length;

this.breadth = breadth;

this.height = height;

}

public float getLength() {

return length;

}

public void setLength(float length) {

this.length = length;

}

public float getBreadth() {

return breadth;

}

public void setBreadth(float breadth) {

this.breadth = breadth;

}

public float getHeight() {

return height;

}

public void setHeight(float height) {

this.height = height;

}

public float volume(){

return length\*breadth\*height;

}

//method to take and return arguments as objects PART (C)

public Box addBox(Box box2){

float len=box2.getLength()+this.length;

float bre=box2.getBreadth()+this.breadth;

float hei=box2.getHeight()+this.height;

return new Box(len,bre,hei);

}

@Override

public String toString() {

return "Length:"+length+" Breadth:"+breadth

+ " Height:"+height; //To change body of generated methods, choose Tools | Templates.

}

// method to demonstrate varargs

public static void displayBoxes(Box...arg){

int i=1;

for(Box box:arg){

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

System.out.println(box.toString());

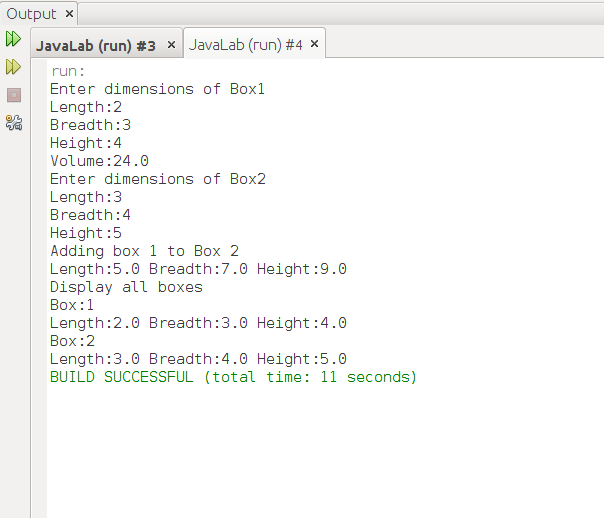
i++;

}

}

}

**OUTPUT**

****

**Experiment 4-(a)** Write a program to give the demo of ‘static’ and ‘final’ variable using Nested class concept.

package javalab;

public class Exp4A {

public static void main(String args[]){

TestOuter1.Inner obj=new TestOuter1.Inner();

obj.msg();

obj.msg2();

}

}

class TestOuter1{

static int data=30;

static final int data2=50;

static class Inner{

void msg(){

System.out.println("static data is "+data);

}

void msg2(){

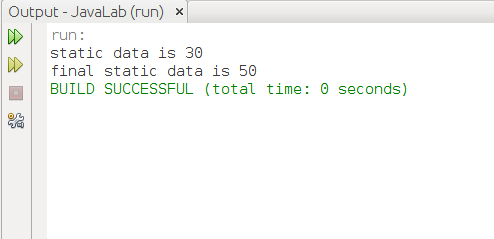
System.out.println("final static data is "+ data2 );

}

}

}

**Output**



**Experiment 4-(b)** Write a program to input command line arguments.

package javalab;

public class Exp4C {

public static void main(String[] args) {

if (args == null) {

System.out.println("enter arguments at command line");

} else {

System.out.println("Command line arguments are");

for (String str : args) {

System.out.println(str);

}

}

}

}

**Output**

