**Fall-2020 Semester**

**Semester**



Submitted by:

Nitish Jaswal (300181850)

BCIS, 2020

Date of Submission:

25-5-2020

Due Date of Submission:

25-5-2020

Submitted to:

Prof. Siddhartha Shyam Vyas

**COMP 155 (Object-Oriented Programming)**

**Java Coding**

**Assignment1**

Table of Contents

[CERTIFICATE OF ORIGINALITY i](#_Toc54085288)

[ACKNOWLEDGEMENT ii](#_Toc54085289)

[COMPANY OVERVIEW & COMPANY INTERACTIONS iii](#_Toc54085290)

[1. Define a class OPERATION with the following specifications: 1](#_Toc54085291)

[2. Define a class BOOK with the following specifications: 10](#_Toc54085292)

[3. Define a class WORKER with the following specifications: 19](#_Toc54085293)

[4. Write a program to find area of Square, Rectangle, and Circle using ‘method overloading’. 24](#_Toc54085294)

[5. Create a class ADD with the following specifications: 33](#_Toc54085295)

[6. Create a class PROD with the following specifications: 38](#_Toc54085296)

[7. Create a class FACT with the following specifications: 43](#_Toc54085297)

[8. Create a class FIB with the following specifications: 48](#_Toc54085298)

[9. Create a class PROGLOGIC with the following specifications: 53](#_Toc54085299)

[Q10. Create a class SCORE with the following specifications: 1](#_Toc54085300)

[References 9](#_Toc54085301)

# CERTIFICATE OF ORIGINALITY

I hereby declare that:

* I have used my own words
* I have not copied
* I have not plagiarized
* I have cited all the sources from where I have adapted / sourced.

**Nitish Jaswal (300181850)**

**Signature**



# ACKNOWLEDGEMENT

**I want to express my profound gratefulness to Prof. Siddhartha Shyam Vyas, Ph.D**

**Nitish Jaswal (300181850)**

**Signature**



# COMPANY OVERVIEW & COMPANY INTERACTIONS

**Professional Name:** Mr. Himanshu Gusain

**Position:** Freelancer

**The platform of Interaction:** Over WhatsApp messaging app

**Contact :** email – himanshugsn30@gmail.com

Ph no- +91 8800950205

**The Interaction is as follow**:

**Que 1:What framework do you use ?**

Reply:

Well, I am a ruby on rails web developer, currently freelancing, other than that I have worked with coding ninjas.

**Que 2: What is ruby on rails, and how you reachout to clients and manage your code ?**

Reply:

Well, The Rails framework helps developers to build websites and applications, sometimes I go for client hunting online , looking for small businesses whose webstie has security, or User interface flaws and call them directly and tell them that I can improve their website, other than that I do bid on the freelancing websites for the ads posted there, and the organization of code is done on the hosting website, I normally suggest hostinger, however I do use git version control sometimes if the web application has too many devlopment phases.

**Que3: Do you use java as developer?**

Reply:

Yes, I do , sometimes clients ask for a android app to be developed, so I use java for that purpose , however iam shifting toward kotlin , as there not much performace difference between the two languages.

Que4: Why do you freelance instead of working for a specific company ?

Ans – what most of the web or mobile application development companies do is that they take contract work in India and assign those projects to employees, so the company takes some part of the money on the application you develop, also sometimes you don’t even get assigned a project. So freelancing increases your knowledge and removes the middleman, if you are a good communicator you can get projects as freelancer easily.

# Define a class OPERATION with the following specifications:

Private members of the class OPERATION are:   
A - integer type   
B - integer type   
Public members of the class OPERATION are:   
Create a default constructor to give the initial values to data members of the class OPERATION   
INPUT() - A function to take A and B as user inputs  
SUM() - A function to return the addition of two numbers DIFF() - A function to return the difference of two numbers  
PROD() - A function to return the product of two numbers  
DIV() - A function to return the quotient of two numbers  
A separate class CALCULATOR shall be created to include the main()  
Also, a MENU-DRIVEN program should be created and the user must continue as long as they want.

**FORM 1 : Acc to Que**

import java.util.\*;

public class Calculator{

public static void main(String[] args) {

Operation op = new Operation();

Scanner sc = new Scanner(System.in);

int option = 0,ex=0;

do

{

System.out.println("Enter your choice from the following menu:");

System.out.println("1.Addition 2.Subtraction

3.Multiplication 4.Division 5.Exit");

option = sc.nextInt();

if(option!=5){

op.INPUT();

}

else

{

break;

}

switch(option)

{

case 1:System.out.println("Addition is "+op.SUM());

break;

case 2:System.out.println("Subtraction is "+op.DIFF());

break;

case 3:System.out.println("Product of is "+op.PROD());

break;

case 4:

System.out.println(" Division is "+op.DIV());

break;

case 5: break;

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

}

System.out.println("Do you want to continue?1.Yes 2.No");

ex=sc.nextInt();

}while(ex==1);

}

}

class Operation{

private int A,B;

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

public Operation(){

A=0;

B=0;

}

public void INPUT(){

System.out.println("Enter the first number");

A=sc.nextInt();

System.out.println("Enter the second number");

B=sc.nextInt();

}

public int SUM(){

return A+B;

}

public int DIFF(){

return A-B;

}

public int PROD(){

return A\*B;

}

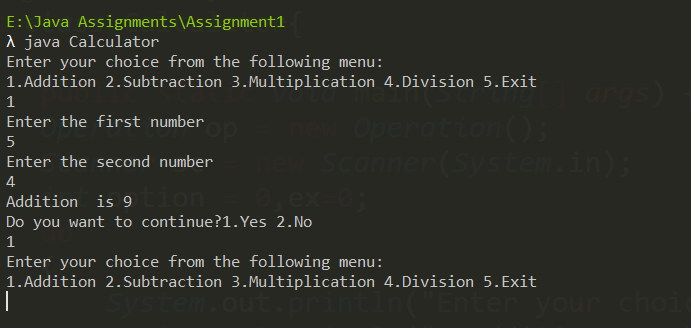
public int DIV(){

return A/B;

}

}

**OUTPUT:**



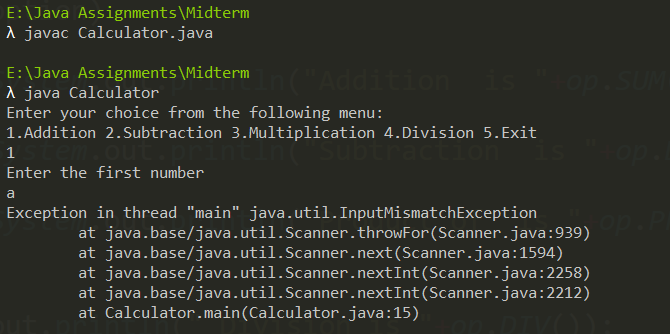
The Program Ask user if he wants to continue or not

Menu is created, to give user choice

DrawBacks

* The program could use static members to improve memory for private members of operation class
* The program could be improved with exception handling

Using try and catch to counter invalid user input



Our Program crashes over invalid input

* This form doesn’t Use This operator to pass the instance , that makes our code more readable instead of passing the member agains and again

FORM 2: Modified

import java.util.\*;

Imported InputMismatchException class for exception handling

import java.util.InputMismatchException;

public class Calculator{

public static void main(String[] args) {

Operation op = new Operation();

Scanner sc = new Scanner(System.in);

int a,b,option = 0,ex=0;

do

{

try{

System.out.println("Enter your choice from the following menu:");

System.out.println("1.Addition 2.Subtraction 3.Multiplication 4.Division 5.Exit");

option = sc.nextInt();

if(option!=5){

System.out.println("Enter the first number");

a=sc.nextInt();

System.out.println("Enter the second number");

b=sc.nextInt();

op.INPUT(a,b);

}

else

{

break;

}

}

catch(InputMismatchException e){

System.out.println("Enter a integer");

}

For clearing the buffer after getting the error

sc.nextLine();

switch(option)

{

case 1:System.out.println("Addition is "+op.SUM());

break;

case 2:System.out.println("Subtraction is "+op.DIFF());

break;

case 3:System.out.println("Product of is "+op.PROD());

break;

case 4:

System.out.println(" Division is "+op.DIV());

break;

case 5: break;

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

}

System.out.println("Do you want to continue?1.Yes 2.No");

ex=sc.nextInt();

}while(ex==1);

}

}

Static members to improve space complexity

class Operation{

private static int A,B;

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

public Operation(){

A=0;

B=0;

}

public void INPUT(int a,int b){

this.A=a;

this.B=b;

}

public int SUM(){

return A+B;

}

public int DIFF(){

return A-B;

}

public int PROD(){

return A\*B;

}

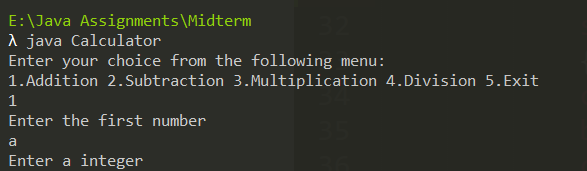
public int DIV(){

return A/B;

}

}

OUTPUT



Error given to the user for entering character

# Define a class BOOK with the following specifications:

Private members of the class BOOK are:   
BOOK\_NO - integer type  
BOOK\_TITLE - string type  
PRICE - float (price per copy)  
TOTAL\_COST() - A function to calculate the total cost for N number of copies, where N is passed to the function as argument   
Public members of the class BOOK are:  
INPUT() - A function to accept values passed as actual parameters from main() to its formal parameters. It further assigns values to private data members of the class BOOK.  
PURCHASE() - Function to invoke TOTAL\_COST() and prints the total cost to be paid by the user   
Create a default constructor to give the initial values to data members of the class BOOK   
A separate class LIBRARY shall be created to include the main()  
main() - must take book\_no, book\_title, price as user input. Also, it shall ask the user to input the number of copies to be purchased.

**Form1:**Acc to Que

import java.util.\*;

public class Library{

public static void main(String[] args) {

Book bk = new Book();

Scanner sc = new Scanner(System.in);

int book\_no=0, N=0, option = 0,ex=0;

float price=0;

String book\_title;

do

{

System.out.println("Enter your choice from the following menu:");

System.out.println("1.Shop 2.Exit");

option = sc.nextInt();

if(option!=5){

System.out.println("Enter the number of copies purchased:");

N = sc.nextInt();

System.out.println("Enter the Book Number");

book\_no=sc.nextInt();

sc.nextLine();

System.out.println("Enter the Book Title");

book\_title=sc.nextLine();

System.out.println("Enter the Book Price");

price=sc.nextFloat();

bk.INPUT(book\_no,book\_title,price);

}

else

{

break;

}

switch(option)

{

case 1:bk.PURCHASE(N);

break;

case 5: break;

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

}

System.out.println("Do you want to continue?1.Yes 2.No");

ex=sc.nextInt();

}while(ex==1);

}

}

class Book{

private int BOOK\_NO;

private String BOOK\_TITLE;

private float PRICE;

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

public Book(){

BOOK\_NO = 0;

PRICE = 0;

BOOK\_TITLE = null;

}

private float TOTAL\_COST(int n){

return PRICE\*n;

}

public void INPUT(int book\_no,String book\_title,float price){

BOOK\_NO = book\_no;

BOOK\_TITLE = book\_title;

PRICE = price;

}

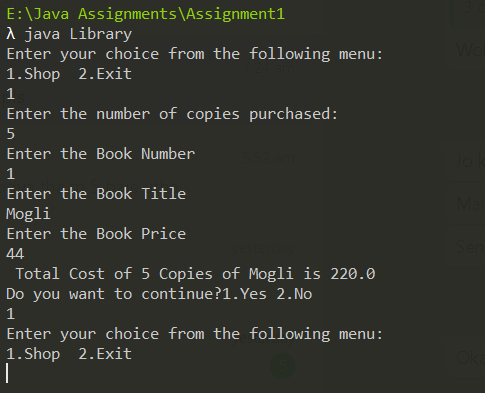
public void PURCHASE(int N){

System.out.println(" Total Cost of "+N+" Copies of "+BOOK\_TITLE+" is "+ TOTAL\_COST(N));

}

}

**OUTPUT:**

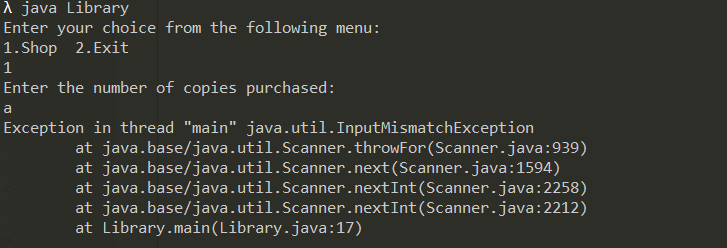


The Program Ask user if he wants to continue or not

Menu is created, to give user choice

**DrawBack**

* The program could use static members to improve memory for private members of operation class
* The program could be improved with exception handling Using try and catch to counter invalid user input.



Our Program crashes over invalid input

* This form doesn’t Use This operator to pass the instance , that makes our code more readable instead of passing the member agains and again**.**

Form 2 - Modified

import java.util.\*;

import java.util.InputMismatchException;

public class Library{

public static void main(String[] args) {

Book bk = new Book();

Scanner sc = new Scanner(System.in);

int book\_no=0, N=0, option = 0,ex=0;

float price=0;

String book\_title;

do

{

try{

System.out.println("Enter your choice from the following menu:");

System.out.println("1.Shop 2.Exit");

option = sc.nextInt();

if(option!=5){

System.out.println("Enter the number of copies purchased:");

N = sc.nextInt();

System.out.println("Enter the Book Number");

book\_no=sc.nextInt();

sc.nextLine();

System.out.println("Enter the Book Title");

book\_title=sc.nextLine();

System.out.println("Enter the Book Price");

price=sc.nextFloat();

bk.INPUT(book\_no,book\_title,price);

}

else

{

break;

}

}

catch(InputMismatchException e){

System.out.println("Enter a integer");

}

sc.nextLine();

switch(option)

{

case 1:bk.PURCHASE(N);

break;

case 5: break;

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

}

System.out.println("Do you want to continue?1.Yes 2.No");

ex=sc.nextInt();

}while(ex==1);

}

}

class Book{

private static int BOOK\_NO;

private static String BOOK\_TITLE;

private static float PRICE;

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

public Book(){

this.BOOK\_NO = 0;

this.PRICE = 0;

this.BOOK\_TITLE = null;

}

private float TOTAL\_COST(int n){

return PRICE\*n;

}

public void INPUT(int book\_no,String book\_title,float price){

this.BOOK\_NO = book\_no;

this.BOOK\_TITLE = book\_title;

this.PRICE = price;

}

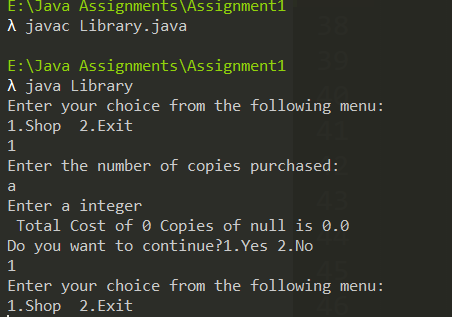
public void PURCHASE(int N){

System.out.println(" Total Cost of "+N+" Copies of "+BOOK\_TITLE+" is "+ TOTAL\_COST(N));

}

}

**OUTPUT**



Error given to the user for entering character

# Define a class WORKER with the following specifications:

Private members of the class WORKER are:  
WNO - integer type  
WNAME - string type  
HRWRK, WGRATE - float (hours worked and wage rate per hour)  
TOTWAGE – float  
CALCWG() - A function to find HRWRK\*WGRATE with float return type  
Public members of the class WORKER are:  
IN\_DATA() - A function to accept values for WNO, WNAME, HRWRK, WGRATE, and invoke CALCWG() to calculate TOTWAGE  
OUT\_DATA() - A function to display all data members on the screen  
Create a default constructor to give the initial values to data members of the class WORKER  
A separate class MANAGER shall be created to include the main().

FORM 1: Acc to Que

import java.util.\*; //imported the util package classes as we need to use Scanner class

public class Manager{

public static void main(String[] args) {

Worker wk = new Worker(); //created object for worker class

wk.IN\_DATA();

wk.OUT\_DATA();

}

}

class Worker{

private int WNO;

private String WNAME;

private float HRWRK, WGRATE,TOTWAGE;

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

private float CALCWG(){

TOTWAGE=HRWRK\*WGRATE;

return TOTWAGE;

}

public Worker(){

WNO=0;

WNAME = null;

HRWRK = 0;

WGRATE = 0;

TOTWAGE = 0;

}

public void IN\_DATA(){

System.out.println("Enter the Worker Number");

WNO=sc.nextInt();

sc.nextLine();

System.out.println("Enter the Worker Name");

WNAME=sc.nextLine();

System.out.println("Enter the Hours worked");

HRWRK=sc.nextFloat();

System.out.println("Enter the hourly Wage rate");

WGRATE=sc.nextFloat();

}

public void OUT\_DATA(){

System.out.println("Worker Number : "+WNO);

System.out.println("Worker Name : "+WNAME);

System.out.println("Hours Worked : "+HRWRK);

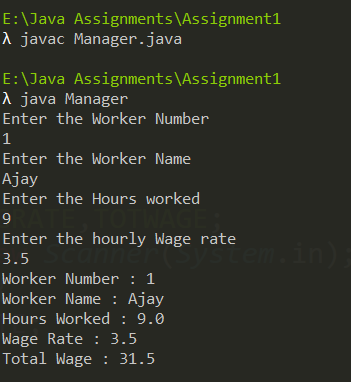
System.out.println("Wage Rate : "+WGRATE);

System.out.println("Total Wage : "+CALCWG());

}

}

**OUTPUT:**



DrawBacks -

* The program could use static members to improve memory for private members of operation class
* The program could be improved with exception handling Using try and catch to counter invalid user input.
* This form doesn’t Use This operator to pass the instance , that makes our code more readable instead of passing the member agains and again**.**

**Note- Unlike previous programs here will do exception handling with scanner class itself, by restricting the buffer input from keyboard. Using hasNext() function in scanner class, the hasNext() function return true if the input matches the token passed as parameter.**

FORM 2: Modified

import java.util.\*; //imported the util package classes as we need to use Scanner class

public class Manager{

public static void main(String[] args) {

Worker wk = new Worker(); //created object for worker class

wk.IN\_DATA();

wk.OUT\_DATA();

}

}

class Worker{

private int WNO;

private String WNAME;

private float HRWRK, WGRATE,TOTWAGE;

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

private float CALCWG(){

TOTWAGE=HRWRK\*WGRATE;

return TOTWAGE;

}

public Worker(){

WNO=0;

WNAME = null;

HRWRK = 0;

WGRATE = 0;

TOTWAGE = 0;

}

public void IN\_DATA(){

System.out.println("Enter the Worker Number");

WNO=sc.nextInt();

sc.nextLine();

System.out.println("Enter the Worker Name");

WNAME=sc.nextLine();

System.out.println("Enter the Hours worked");

HRWRK=sc.nextFloat();

System.out.println("Enter the hourly Wage rate");

WGRATE=sc.nextFloat();

}

public void OUT\_DATA(){

System.out.println("Worker Number : "+WNO);

System.out.println("Worker Name : "+WNAME);

System.out.println("Hours Worked : "+HRWRK);

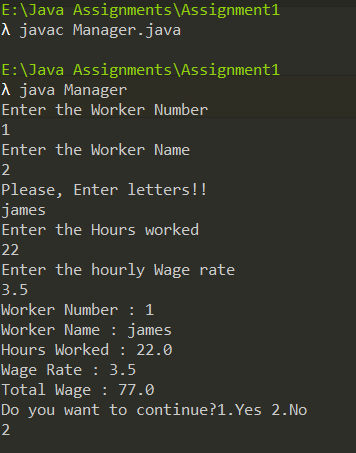
System.out.println("Wage Rate : "+WGRATE);

System.out.println("Total Wage : "+CALCWG());

}

}

**OUTPUT:**



Error given during the flow of our program, for entering number instead of name

# Write a program to find area of Square, Rectangle, and Circle using ‘method overloading’.

Private members of the class DIFFERENT\_SHAPES are:

LENGTH - float type

BREATH - float type

SIDE - float type

RADIUS - double type

Public members of the class DIFFERENT\_SHAPES are:

Create a default constructor to give the initial values to data members of the class DIFFERENT\_SHAPES

AREA() - A function to assign values to the respective private data members and return the area of a circle

AREA() - A function to assign values to the respective private data members and return the area of a rectangle

AREA() - A function to assign values to the respective private data members and return the area of a square

A separate class SHAPES shall be created to include the main()

main() - must take all the user inputs like: side, radius, length, breadth

A MENU-DRIVEN program should be created and a user input is taken with respect to area of which shape must be computed, and then depending upon user’s choice, that overloaded AREA() method shall be called depending upon the shape.

FORM 1: acc to que

import java.util.\*;

public class Shapes{

public static void main(String args[])

{

DIFFERENT\_SHAPES ds = new DIFFERENT\_SHAPES();

Scanner sc = new Scanner(System.in);

int option = 0,ex=0;

float side, length, breath;

double radius;

do

{

System.out.println("Enter your choice from the following menu:");

System.out.println("1.Square 2.Rectangle 3.Circle 5.Exit");

option = sc.nextInt();

switch(option)

{

case 1: System.out.print("Enter the side :");

side = sc.nextFloat();

System.out.println("Area of Square is "+ds.area(side));

break;

case 2: System.out.print("Enter the length :");

length = sc.nextFloat();

System.out.print("\nEnter the breath :");

breath = sc.nextFloat();

System.out.println("Area of Rectangle is "+ds.area(length,breath));

break;

case 3: System.out.print("Enter the radius :");

radius = sc.nextDouble();

System.out.println("Area of Circle is "+ds.area(radius));

break;

case 5: break;

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

}

System.out.println("Do you want to continue?1.Yes 2.No");

ex=sc.nextInt();

}while(ex==1);

}

}

class DIFFERENT\_SHAPES

{

private float LENGTH,BREATH,SIDE;

private double RADIUS;

public DIFFERENT\_SHAPES(){

LENGTH=0;

BREATH=0;

SIDE=0;

RADIUS=0;

}

public float area(float x)

{

return x\*x;

}

public float area(float x, float y)

{

return x\*y;

}

public double area(double x)

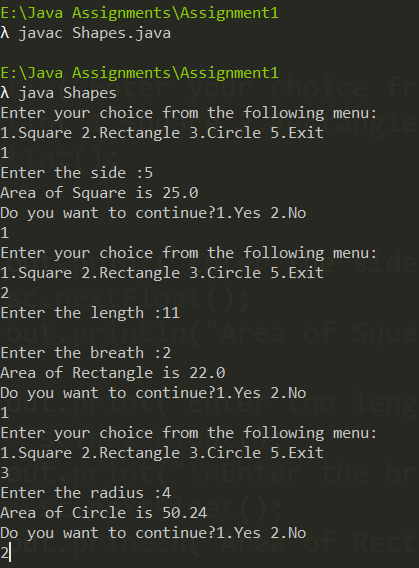
{

return 3.14\*x\*x;

}

}

OUTPUT**:**



Form 2 - Modified

import java.util.\*;

import java.util.InputMismatchException;

public class Shapes{

public static void main(String args[])

{

DIFFERENT\_SHAPES ds = new DIFFERENT\_SHAPES();

Scanner sc = new Scanner(System.in);

int option = 0,ex=0;

float side, length, breath;

double radius;

do

{

try{

System.out.println("Enter your choice from the following menu:");

System.out.println("1.Square 2.Rectangle 3.Circle 5.Exit");

option = sc.nextInt();

switch(option)

{

case 1: System.out.print("Enter the side :");

side = sc.nextFloat();

System.out.println("Area of Square is "+ds.area(side));

break;

case 2: System.out.print("Enter the length :");

length = sc.nextFloat();

System.out.print("\nEnter the breath :");

breath = sc.nextFloat();

System.out.println("Area of Rectangle is "+ds.area(length,breath));

break;

case 3: System.out.print("Enter the radius :");

radius = sc.nextDouble();

System.out.println("Area of Circle is "+ds.area(radius));

break;

case 5: break;

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

}

}

catch(InputMismatchException e){

System.out.println("Enter a integer");

}

sc.nextLine();

System.out.println("Do you want to continue?1.Yes 2.No");

ex=sc.nextInt();

}while(ex==1);

}

}

class DIFFERENT\_SHAPES

{

Private static float LENGTH,BREATH,SIDE;

private double RADIUS;

public DIFFERENT\_SHAPES(){

LENGTH=0;

BREATH=0;

SIDE=0;

RADIUS=0;

}

public float area(float x)

{

return x\*x;

}

public float area(float x, float y)

{

return x\*y;

}

public double area(double x)

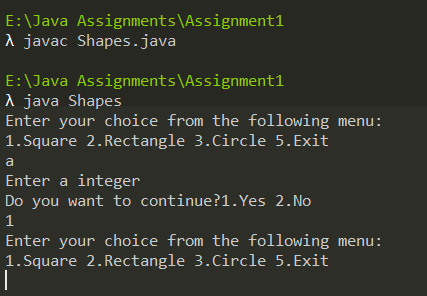
{

return 3.14\*x\*x;

}

}

OUTPUT -



Error given to the user for entering character

# Create a class ADD with the following specifications:

Private members:

A - integer type

B - integer type

Public members:

Create a default constructor to give the initial values to data members of the class ADD

INPUT() - is a function to take A and B as a user input

COMPUTE\_SUM() - is a function that will compute the sum of two numbers and return it to the main().

A separate class ADDITION shall be created to include the main()

main() - must invoke INPUT() and then COMPUTE\_SUM() to print the sum of two numbers

The program must continue as long as the user wants.

Form 1: acc to que

import java.util.\*;

class Addition{

public static void main(String[] args) {

ADD add = new ADD();

Scanner sc = new Scanner(System.in);

int ex=0;

do{

add.INPUT();

System.out.println("Sum is :"+add.COMPUTESUM());

System.out.println("Do you want to continue?1.Yes 2.No");

ex=sc.nextInt();

}while(ex==1);

}

}

class ADD{

private int A,B;

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

public ADD(){

A=0;

B=0;

}

public void INPUT(){

System.out.println("Enter the first number");

A=sc.nextInt();

System.out.println("Enter the second number");

B=sc.nextInt();

}

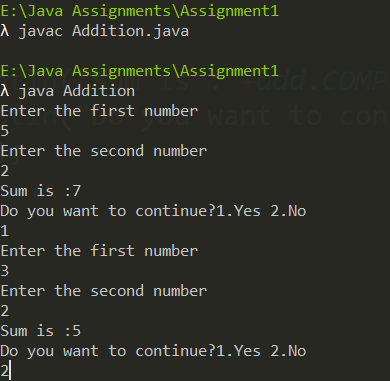
public int COMPUTESUM(){

return A+B;

}

}

OUTPUT –



DrawBacks -

* The program could use static members to improve memory for private members of operation class
* The program could be improved with exception handling Using try and catch to counter invalid user input.
* We will also a septrate function for exception handling, I named it **valid**

**Note- here also will do exception handling with scanner class itself, by restricting the buffer input from keyboard. Using hasNext() function in scanner class, the hasNext() function return true if the input matches the token passed as parameter.**

FORM 2: Modified

import java.util.\*;

class Addition{

public static void main(String[] args) {

ADD add = new ADD();

Scanner sc = new Scanner(System.in);

int ex=0;

do{

add.INPUT();

System.out.println("Sum is :"+add.COMPUTESUM());

System.out.println("Do you want to continue?1.Yes 2.No");

ex=sc.nextInt();

}while(ex==1);

}

}

class ADD{

private static int A,B;

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

public ADD(){

A=0;

B=0;

}

public void INPUT(){

System.out.println("Enter the first number");

valid();

A=sc.nextInt();

System.out.println("Enter the second number");

valid();

B=sc.nextInt();

}

public int COMPUTESUM(){

return A+B;

}

public void valid(){

while (!sc.hasNext("[0-9]+")) {

System.out.println("Please, Enter numbers!!");

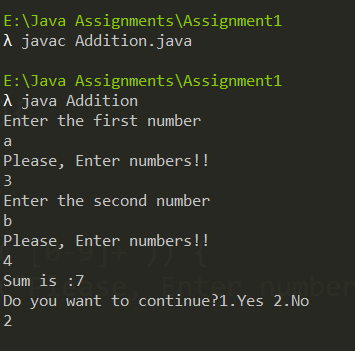
sc.next();

}

}

}

**OUTPUT:**



Error given

# Create a class PROD with the following specifications:

Private members:

A – integer type

B – integer type

Public members:

Create a default constructor to give the initial values to data members of the class PROD

INPUT() – a function to recieve user inputs as arguments in its formal parameters from actual parameters in main(). It must further assign those values to the private data members of PROD class

COMPUTE\_PROD() – is a function that will compute the product of two numbers and return it to the main()

A separate class PRODUCT shall be created to include the main()

main() – must take A and B as user input and invoke INPUT() and then further invoke COMPUTE\_PROD()

The program must continue as long as the user wants.

Form 1: acc to que

import java.util.\*;

class Product{

public static void main(String[] args) {

PROD pd = new PROD();

Scanner sc = new Scanner(System.in);

int ex=0;

do{

pd.INPUT();

System.out.println("Product is :"+pd.COMPUTEPROD());

System.out.println("Do you want to continue?1.Yes 2.No");

ex=sc.nextInt();

}while(ex==1);

}

}

class PROD{

private int A,B;

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

public PROD(){

A=0;

B=0;

}

public void INPUT(){

System.out.println("Enter the first number");

A=sc.nextInt();

System.out.println("Enter the second number");

B=sc.nextInt();

}

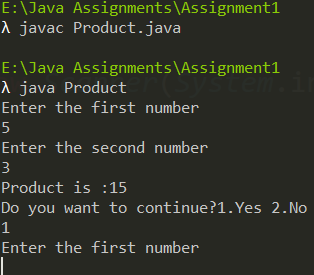
public int COMPUTEPROD(){

return A\*B;

}

}

OUTPUT-



DrawBacks -

* The program could use static members to improve memory for private members of operation class
* **Program is not made for taking series of inputs, so now we are going to do a completely different setup by accessing the iputs for A and B from a text file.**

**Note- here we are taking input from a file so, the only exception handling we had to is to check if the file exists. We will use delimeter method of Scnner class to read and create a file object from File class of java.io package.**

FORM 2: Modified

import java.util.\*;

import java.io.\*;

class Product{

public static void main(String[] args) {

PROD pd = new PROD();

Scanner s = new Scanner(System.in);

int ex=0,a,b;

do{

try{

File file = new File(args[0]);

Scanner sc = new Scanner(file);

sc.useDelimiter("\\D+");

while(sc.hasNext()){

a = sc.nextInt();

Reason for commenting this is below output

b = sc.nextInt();

//sc.nextLine();

pd.INPUT(a,b);

System.out.println("1st number is =" + a + "\n2nd number is =" + b);

System.out.println("Product is :"+pd.COMPUTEPROD());

}

}

catch(FileNotFoundException e){

System.out.println("File doesnt exist!!");

}

System.out.println("Do you want to continue?1.Yes 2.No");

ex=s.nextInt();

}while(ex==1);

}

}

class PROD{

private int A,B;

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

public PROD(){

A=0;

B=0;

}

public void INPUT(int a,int b){

A=a;

B=b;

}

public int COMPUTEPROD(){

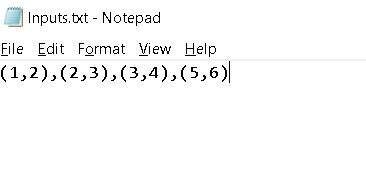
return A\*B;

}

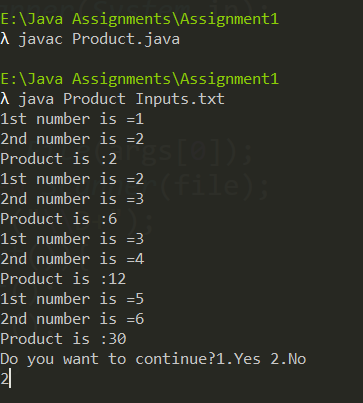
}

**OUTPUT:**

Our Text file for taking input -



Note - As, you can see we somewhat have followed a format of a CSV file ( comma seprated ) , this is because of the use of useDelimeter() function, another reason is its more easy to read from a csv file rather than randomly accessing.



Taking series of input from the text file named “Inputs”

Note - You might have noticed after inputing from the object of scanner class who is reading from file, I commented sc.nextLine() which we previously used to clear the buffer, however I figured out that it was exausting more memory, and making reading process slower as the scanner now have to flush the buffer every time it reads, and also this function is already being done by our hasNext() function, so I left the statement there to mention this.

# Create a class FACT with the following specifications:

Private members:

fct - integer type in which the factorial will be accumulated

Public members:

Create a default constructor to give the initial values to data members of the class FACT

COMPUTE\_FACT() - is a function that will compute the factorial of an integer and return it to the main()

A separate class FACTORIAL shall be created to include the main()

main() - must take the N as a user input of which the factorial needs to be computed and invoke COMPUTE\_FACT()

The program must continue as long as the user wants.

FORM 1:-

import java.util.\*;

class Factorial{

public static void main(String[] args) {

FACT fc = new FACT();

Scanner sc = new Scanner(System.in);

int ex=0,N=0;

do{

System.out.println("Enter the number to get the factorial: ");

N=sc.nextInt();

System.out.println("Factorial of "+N+" is :"+fc.COMPUTE\_FACT(N));

System.out.println("Do you want to continue?1.Yes 2.No");

ex=sc.nextInt();

}while(ex==1);

}

}

class FACT{

private int fct;

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

public FACT() {

fct = 1;

}

public int COMPUTE\_FACT(int N) {

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

fct = fct\*i;

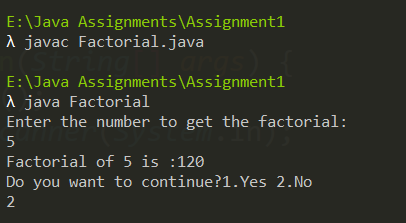
}

return fct;

}

}

OUTPUT-



DrawBacks -

* The program could use static members to improve memory for private members of operation class
* **Program is not made for taking series of inputs, so like from previous example, we take input form a text file.**
* **We can do factorial program from various ways, we can do it recursively ( not so optimal) , or iteratively that we have done in our previous form, so in case of performance of our logic our program is optimal, however there another way to do that is ternary recursive method which reduces the code in our factorial method.**

**Note- here we are taking input from a file so, the only exception handling we had to is to check if the file exists. We will use delimeter method of Scnner class to read and create a file object from File class of java.io package.**

FORM 2: Modified

import java.util.\*;

import java.io.\*;

class Factorial{

public static void main(String[] args) {

FACT fc = new FACT();

Scanner s = new Scanner(System.in);

int ex=0,N=0;

do{

try{

File file = new File(args[0]);

Scanner sc = new Scanner(file);

sc.useDelimiter("\\D+");

while(sc.hasNext()){

N = sc.nextInt();

System.out.println("number is =" + N) ;

System.out.println("Factorial of "+N+" is :"+fc.COMPUTE\_FACT(N));

}

}

catch(FileNotFoundException e){

System.out.println("File doesnt exist!!");

}

System.out.println("Do you want to continue?1.Yes 2.No");

ex=s.nextInt();

}while(ex==1);

}

}

class FACT{

I have left the memebers of FACT class which became useles with the implementation of ternary recursion method.  
now we don’t need fct

// private int fct;

// public FACT() {

// fct = 1;

// }

public int COMPUTE\_FACT(int N) {

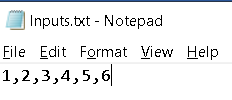
return (N == 1 || N == 0) ? 1 : N \* COMPUTE\_FACT(N - 1);

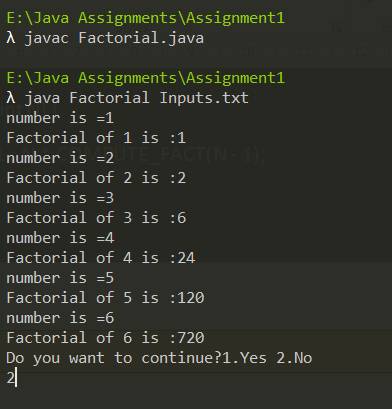
}

}

**OUTPUT:**

Our Text file for taking input -





Taking series of input from the text file named “Inputs”

Note – As we are taking input from file the time and space complexity will totally depend on the size of file, however we have used two loops in our main function we can assume the base case of complexity will be around O(n2).

# Create a class FIB with the following specifications:

Private members:

first - integer type

second - integer type

third - integer type

Public members:

Create a default constructor to give the initial values to data members of the class FIB

GENERATE\_FIB() - is a function that will print the fibonacci series upto the user limit

A separate class FIBONACCI shall be created to include the main()

main() - must take the N (series limit) as a user input and invoke GENERATE\_FIB()

The program must continue as long as the user wants..

FORM 1:-

import java.util.\*;

class Fibonacci{

public static void main(String[] args) {

int N=0,ex=0;

FIB f = new FIB();

Scanner sc = new Scanner(System.in);

do

{

System.out.println("Enter the fibonacci series limit: ");

N = sc.nextInt();

f.GENERATE\_FIB(N);

System.out.println("\nDo you want to continue?1.Yes 2.No");

ex=sc.nextInt();

}while(ex==1);

}

}

class FIB{

private int FIRST, SECOND, THIRD;

public FIB() {

FIRST = 0;

SECOND = 1;

THIRD=0;

}

public void GENERATE\_FIB(int N) {

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

System.out.print(FIRST + " ");

THIRD = FIRST + SECOND;

FIRST = SECOND;

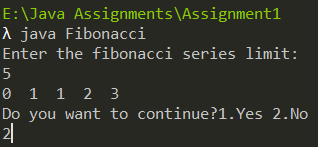
SECOND = THIRD;

}

}

}

OUTPUT-



DrawBacks -

* The program could use static members to improve memory for private members of operation class
* **Program is not made for taking series of inputs, so like from previous example, we take input form a text file.**
* **We can do Fibonacci program from various ways, we can do it recursively ( not so optimal) , or iteratively that we have done in our previous form, so in case of performance of our logic our program is optimal,   
  however we are going to use dynamic programming to define our logic for our fibonnaci function.**

**Note- here we are taking input from a file so, the only exception handling we had to is to check if the file exists. We will use delimeter method of Scnner class to read and create a file object from File class of java.io package.**

FORM 2: Modified

import java.util.\*;

import java.io.\*;

class Fibonacci{

public static void main(String[] args) {

int N=0,ex=0;

FIB f = new FIB();

Scanner s = new Scanner(System.in);

do

{

try{

File file = new File(args[0]);

Scanner sc = new Scanner(file);

sc.useDelimiter("\\D+");

while(sc.hasNext()){

N = sc.nextInt();

System.out.println("\nthe fibonacci limit is: "+N);

System.out.print("\nThe fibonacci series is: ");

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

{

System.out.print(f.GENERATE\_FIB(i)+" ");

}

}

}

catch(FileNotFoundException e){

System.out.println("File doesnt exist!!");

}

System.out.println("\nDo you want to continue?1.Yes 2.No");

ex=s.nextInt();

}while(ex==1);

}

}

class FIB{

public int GENERATE\_FIB(int N) {

int fn[] = new int[N + 2];

int i;

fn[0] = 0; fn[1] = 1;

for (i = 2; i <= N; i++){

fn[i] = fn[i - 1] + fn[i - 2];

}

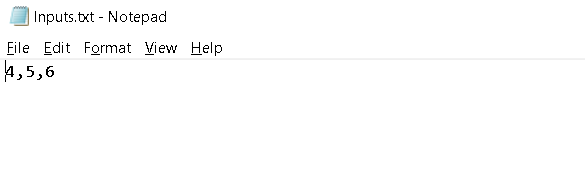
return fn[N];

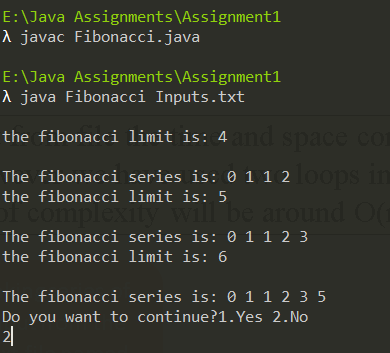
}

}

**OUTPUT:**

Our Text file for taking input -





Taking series of input from the text file named “Inputs”

Note – As we are taking input from file the time and space complexity will totally depend on the size of file, however we have used one loops in our main function that is for passing the value to function to calculate the series we can assume the base case of complexity will be around O(N).

# Create a class PROGLOGIC with the following specifications:

Private members:

N - integer type

Public members:

Create a default constructor to give the initial values to data members of the class PROGLOGIC

INPUT() - is a function that will recieve argument in their formal parameter from main() and assign values to the private data members of PROGLOGIC

FINDING\_EVENODD() - checks whether given user input is an even or odd number and return the value to main() for printing

A separate class EVEN shall be created to include the main()

main() - must invoke INPUT() and then invoke EVEN\_ODD() to print whether the given number is an even or odd

The program must continue as long as the user wants..

FORM 1:-

import java.util.\*;

class Even{

public static void main(String[] args) {

int ex=0;

Scanner sc = new Scanner(System.in);

PROLOGIC p = new PROLOGIC();

do

{

p.INPUT();

p.FINDING\_EVENODD();

System.out.println("\nDo you want to continue?1.Yes 2.No");

ex=sc.nextInt();

}while(ex==1);

}

}

class PROLOGIC{

private int N;

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

public PROLOGIC(){

N=0;

}

public void INPUT(){

System.out.print("\nEnter the number: ");

N=sc.nextInt();

}

public void FINDING\_EVENODD(){

if(N%2==0){

System.out.println(N+" is Even! ");

}

else{

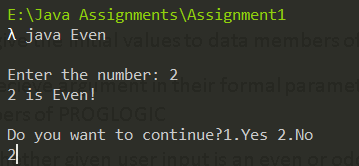
System.out.println(N+" is Odd! ");

}

}

}

OUTPUT-



DrawBacks -

* The program could use static members to improve memory for private members of operation class
* **Program is not made for taking series of inputs, so like from previous example, we take input form a text file.**
* **We can optimize our even function , by making the function as string type and using ternary operators to check the condition and then return it to the main function of the other class, instead of using series of if and else conditions.**

**Note- here we are taking input from a file so, the only exception handling we had to is to check if the file exists. We will use delimeter method of Scnner class to read and create a file object from File class of java.io package.**

FORM 2: Modified

import java.util.\*;

import java.io.\*;

class Even{

public static void main(String[] args) {

int ex=0,N;

Scanner s = new Scanner(System.in);

PROLOGIC p = new PROLOGIC();

do

{

try{

File file = new File(args[0]);

Scanner sc = new Scanner(file);

sc.useDelimiter("\\D+");

while(sc.hasNext()){

N = sc.nextInt();

System.out.println("\nthe number is: "+N);

p.INPUT(N);

System.out.println("The number is "+p.FINDING\_EVENODD());

}

}

catch(FileNotFoundException e){

System.out.println("File doesnt exist!!");

}

System.out.println("\nDo you want to continue?1.Yes 2.No");

ex=s.nextInt();

}while(ex==1);

}

}

class PROLOGIC{

private int N;

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

public PROLOGIC(){

N=0;

}

public void INPUT(int n){

N=n;

Code reduced and ternary operator are more faster than if and else

}

public String FINDING\_EVENODD(){

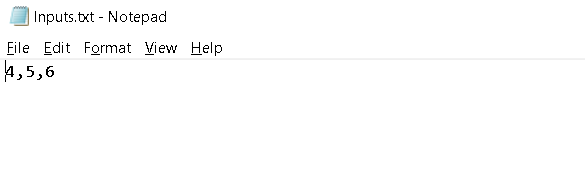
return (N % 2 == 0) ? "even" : "odd";

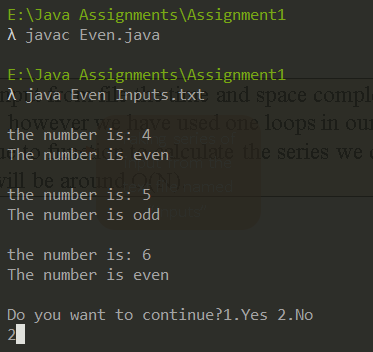
}

}

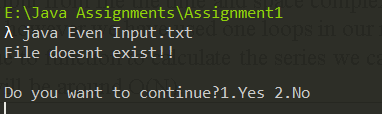
**OUTPUT:**

Our Text file for taking input -





Taking series of input from the text file named “Inputs”



Error gicen if the file doesn’t exist

# Q10. Create a class SCORE with the following specifications:

Private members:

SUB1 - float type

SUB2 - float type

SUB3 - float type

SUB4 - float type

SUB5 - float type

SUM() - is used to sum up the marks in 5 different subjects and return the sum to CALC\_PERC()

Public members:

Create a default constructor to give the initial values to data members of the class SCORE

INPUT() - is a function that will receive argument in their formal parameters from main() and assign values to private data members of SCORE

CALC\_PERC() - is used to invoke SUM() and then compute percentage and return it to CALC\_GRADE()

CALC\_GRADE() - is used to invoke CALC\_PERC() and then compute the final grade and return it to main() for printing

A separate class FINALEXAM shall be created to include the main()

main() - must take scores from the users in 5 different subjects and invoke INPUT(). It must further invoke CALC\_GRADE() to print the final grade

The program must continue as long as the user wants.

FORM 1:- Acc to the question

import java.util.\*;

class FinalExam{

public static void main(String[] args) {

int ex=0;

float sub1=0, sub2=0, sub3=0, sub4=0, sub5=0;

Score s = new Score();

Scanner sc = new Scanner(System.in);

do

{

System.out.println("Enter the Marks of 1st subject :");

sub1 = sc.nextInt();

System.out.println("Enter the Marks of 2nd subject :");

sub2 = sc.nextInt();

System.out.println("Enter the Marks of 3rd subject :");

sub3 = sc.nextInt();

System.out.println("Enter the Marks of 4th subject :");

sub4 = sc.nextInt();

System.out.println("Enter the Marks of 5th subject :");

sub5 = sc.nextInt();

s.INPUT(sub1,sub2,sub3,sub4,sub5);

System.out.println("Final Grade: "+s.CALC\_GRADE());

System.out.println("\nDo you want to continue?1.Yes 2.No");

ex=sc.nextInt();

}while(ex==1);

}

}

class Score{

private float SUB1,SUB2,SUB3,SUB4,SUB5;

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

public Score(){

SUB1=0;

SUB2=0;

SUB3=0;

SUB4=0;

SUB5=0;

}

public void INPUT(float sub1,float sub2,float sub3,float sub4,float sub5){

SUB1 = sub1;

SUB2 = sub2;

SUB3 = sub3;

SUB4 = sub4;

SUB5 = sub5;

}

private float SUM(){

return SUB1+SUB2+SUB3+SUB4+SUB5;

}

public float CALC\_PERC(){

return (SUM()/500)\*100;

}

public char CALC\_GRADE(){

if(CALC\_PERC()>=80){

return 'A';

}

else if(CALC\_PERC()>=60 && CALC\_PERC()<80){

return 'B';

}

else if(CALC\_PERC()>=40 && CALC\_PERC()<60){

return 'C';

}

else {

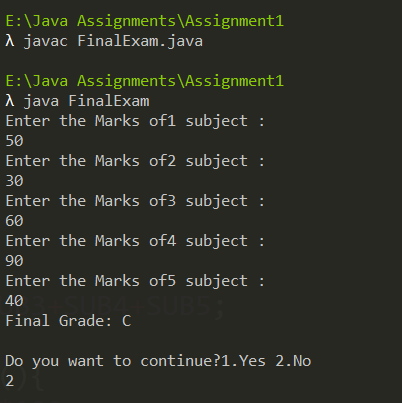
return 'D';

}

}

}

OUTPUT-



DrawBacks -

* The program could use static members to improve memory for private members of operation class
* **Program is not made for taking series of inputs, so like from previous example, we take input form a text file.**
* **We will optimize this program, instead of taking input for each subject we run a loop, to read from file and pass it to an array in our INPUT() function to a dynamically allocated array mrk[].**

**Note- here we are taking input from a file so, the only exception handling we had to is to check if the file exists. We will use delimeter method of Scnner class to read and create a file object from File class of java.io package.**

FORM 2: Modified

import java.util.\*;

import java.io.\*;

class FinalExam{

public static void main(String[] args) {

int ex=0,n=0;

float N=0;

//float sub1=0, sub2=0, sub3=0, sub4=0, sub5=0;

Score sco = new Score();

Scanner s = new Scanner(System.in);

do

{

try{

File file = new File(args[0]);

Scanner sc = new Scanner(file);

sc.useDelimiter("\\D+");

while(sc.hasNext()){

while(n!=5){

N = sc.nextInt();

System.out.println("The Marks of "+(n+1)+" subject is :"+ N);

sco.INPUT(N);

n++;

}

sc.nextLine();

}

}

catch(FileNotFoundException e){

System.out.println("File doesnt exist!!");

}

System.out.println("Final Grade is: "+sco.CALC\_GRADE());

System.out.println("\nDo you want to continue?1.Yes 2.No");

ex=s.nextInt();

}while(ex==1);

}

}

class Score{

private float mrk[] = new float[10];

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

public void INPUT(float N){

Allocating the input from file to the array

for (int i=0; i<5; i++ ) {

if(N<=100){

mrk[i] = N;

}

else{

System.out.println("The file contain invalid input!!");

System.exit(0);

}

}

}

private float SUM(){

float sum=0;

for (int i=0; i<5; i++) {

sum = sum+mrk[i];

}

return sum;

}

public float CALC\_PERC(){

return (SUM()/500)\*100;

}

public char CALC\_GRADE(){

if(CALC\_PERC()>=80){

return 'A';

}

else if(CALC\_PERC()>=60 && CALC\_PERC()<80){

return 'B';

}

else if(CALC\_PERC()>=40 && CALC\_PERC()<60){

return 'C';

}

else {

return 'D';

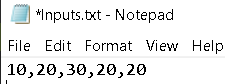
}

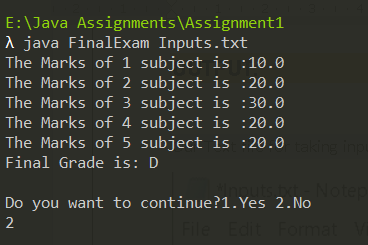
}

}

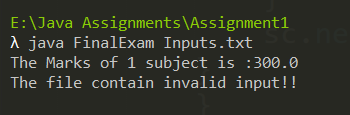
**OUTPUT:**

Our Text file for taking input -

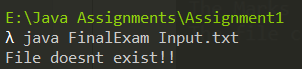




Taking series of input from the text file named “Inputs”



Error given if there are marks greater than 100 in out text file



Error given if the file doesn’t exist

# References

* Parahar(2019, Nov 26). Difference between Scanner and BufferReader Class in Java. Retrieved <https://www.tutorialspoint.com/difference-between-scanner-and-bufferreader-class-in-java> ( 2020,18 Oct)
* JavaTpoint.(2020). Java Scanner hasNext() Method. Retrieved from <https://www.javatpoint.com/post/java-scanner-hasnext-method>   
  (2020,2 Oct)
* StackOverflow.(n.d.). IOException; must be caught or declared to be thrown.Retrieved from <https://stackoverflow.com/questions/17194343/ioexception-must-be-caught-or-declared-to-be-thrown> (2020,17 Oct)
* StackOverflow.(n.d.). Same exception type with different error message.Retrieved from <https://stackoverflow.com/questions/45921034/same-exception-type-with-different-error-message> (2020,17 Oct)
* StackOverflow(n.d.). How can I organize my Java code?.Retrieved from <https://stackoverflow.com/questions/30561710/how-can-i-organize-my-java-cod>