**Fall-2020 Semester**

**Semester**



Submitted by:

Nitish Jaswal (300181850)

BCIS, 2020

Date of Submission:

30-10-2020

Due Date of Submission:

30-10-2020

Submitted to:

Prof. Siddhartha Shyam Vyas

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

**Java Coding**

**Assignment2**

Table of Contents

[CERTIFICATE OF ORIGINALITY i](#_Toc54532418)

[ACKNOWLEDGEMENT ii](#_Toc54532419)

[COMPANY OVERVIEW & COMPANY INTERACTIONS iii](#_Toc54532420)

[1. Create a class POWER with the following specifications: 1](#_Toc54532421)

[2. Create a class PERIMETER with the following specifications: 9](#_Toc54532422)

[3. Create a class EVENODDPRIME with following specifications: 18](#_Toc54532423)

[4. Create a class ARMSTRONG\_NUMBER with following specifications:. 23](#_Toc54532424)

[5. Create a class NATURAL\_NUMBERS with following specifications: 32](#_Toc54532425)

[6. Create a class TABLE with following specifications: 37](#_Toc54532426)

[7. Create a class FINDING\_CHARACTER with following specifications: 42](#_Toc54532427)

[8. Create a class DESCENDING with following specifications: 47](#_Toc54532428)

[9. Create a class REVERSE\_PALINDROME with following specifications: 53](#_Toc54532429)

[Q10. Create a class SUMOFDIGITS with following specifications: 1](#_Toc54532430)

[References 9](#_Toc54532431)

# 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.

**Professional Name:** Mr. Akshay Tyagi

**Company Name – NorthCorpSoftware**

**Company Overview-**

A delhi based firm which help business automate and make their practices more efficient with help of new technologies like IoT

They serve in telecommunication industry, automobile industry and Fast-Moving Consumer Goods firms

**Position:** Software Developer

**The platform of Interaction:** Instagram chat

**Contact :** email – [akshaytyagi@north.tech](mailto:akshaytyagi@north.tech)

Github - <https://github.com/akshay20t>

**The Interaction is as follow**:

**Que 1: What technologies you have worked on?**

I have worked with framework such as Django and CherryPy for python, and even at work I use these technologies, our company has setup its database connection , with Django ORM.

**Que 2:** **Have you done any system design from scratch?**

Yes, I have developed Database Interface to acces and manipulate database based on python GUI. It is developed using pymysql and tkinter modules. You can see my projects on my github.

# Create a class POWER with the following specifications:

Private members:

COUNTER - integer type in which the power of a given number will be accumulated

Public members:

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

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

A separate class MAIN\_POWER shall be created to include the main()

main() - It must take the N as a user input of which the power needs to be calculated.

It must also take PWR as a user input to get the power for calculation

It must further invoke COMPUTE\_POWER() to compute N raised to the power PWR

It must finally print the power of a given number

The program must continue as long as the user wants.

**FORM 1 : Acc to Que**

import java.util.\*;

public class MAIN\_POWER{

public static void main(String[] args) {

//creating obj of scanner class for input

Scanner sc = new Scanner(System.in);

int N=0,PWR=0,ex=0;

POWER pw = new POWER();

do

{

//inputing number and power

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

N=sc.nextInt();

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

PWR=sc.nextInt();

//printing result

System.out.println(N+" ^ "+PWR+" = "+pw.COMPUTE\_POWER(N,PWR));

//asking user if he want to continue

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

ex=sc.nextInt();

}while(ex==1);

}

}

class POWER{

private int COUNTER; //declared counter var

//constructor to initialize members

public POWER(){

COUNTER = 0;

}

public int COMPUTE\_POWER(int n,int pwr){

COUNTER = pwr; //assigning power to counter to run loop

int num=n; //storing the value of number

//loop to compute power

while(COUNTER!=1){

n=num\*n;

COUNTER--;

}

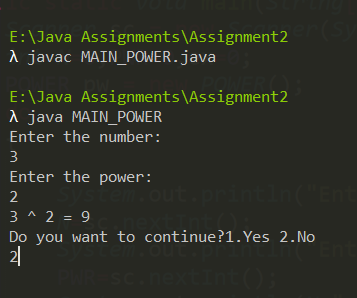
//returning the result to main

return n;

}

}

**OUTPUT:**

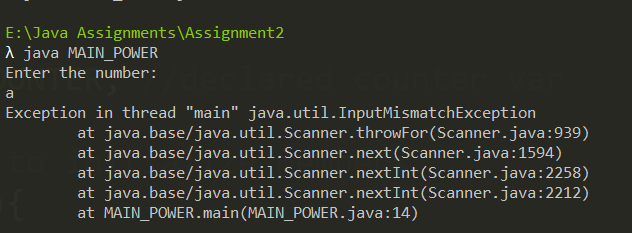


The Program Ask user if he wants to continue or not

DrawBacks

* **The first problem with this program is the reusability of our POWER class, instead of passing it in the function rather passing it through the constructor.**
* **The program could use static members as the memory asigned to static members is fixed so that will help our program to compute inputs faster.**
* **Passign many inputs through a CSV ( comma seprated file) is not possible and it is good to have huge input processing in program so that user can use a data file .**
* **The program could be improved with exception handling**

**Using try and catch to counter invalid user input**



Our Program crashes over invalid input

FORM 2: Modified

import java.util.\*;

Imported InputMismatchException

And FileNotFoundException class for exception handling

import java.io.InputMismatchException;

import java.io.FileNotFoundException;

public class MAIN\_POWER{

public static void main(String[] args) {

//creating obj of scanner class for input

Scanner sc = new Scanner(System.in);

int N=0,PWR=0,ex=0;

do

{

try{

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

Scanner fs = new Scanner(file);

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

while(fs.hasNext()){

N = fs.nextInt();

PWR = fs.nextInt();

POWER pw = new POWER(N,PWR);

System.out.println(N+" ^ "+PWR+" = "+pw.COMPUTE\_POWER());

}

}

catch(InputMismatchException|FileNotFoundException e){

System.out.println("Invalid input!");

For clearing the buffer after getting the error

}

sc.nextLine();

//asking user if he want to continue

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

ex=sc.nextInt();

}while(ex==1);

}

}

class POWER{

private static int COUNTER,NUM; //declared counter var

Static members to improve space complexity

//constructor to initialize members

public POWER(int N, int PWR){

NUM = N;

COUNTER = PWR;

}

public int COMPUTE\_POWER(){

int num=NUM; //storing the value of number

//loop to compute power

while(COUNTER!=1){

NUM=num\*NUM;

COUNTER--;

}

//returning the result to main

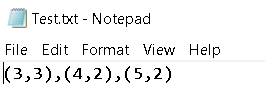
return NUM;

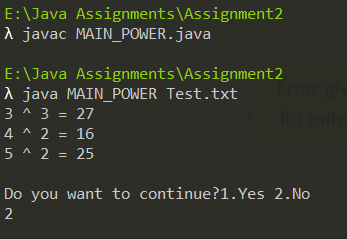
}

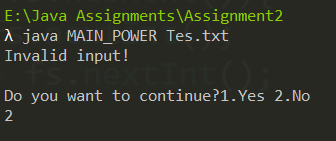
}

OUTPUT

Text File



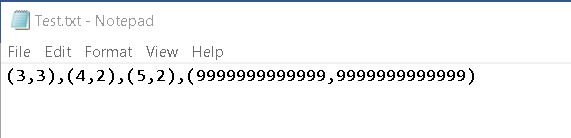




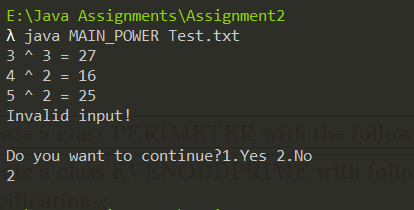
Error given to the user for entering character

**Drawback**

* **The buffer crashes when the input is larger from the file or there are many inputs , and just throws the exception of TypeMismatch**



Larger input



Exception Thrown, However our program didn’t crash which is a good thing, now we handle this issue

**So now we have to improve our current program flow , and data types to handle the large input.**

**NOTE:- For this purpose we are going to use BIGINTEGER class, which can take huge inputs and process them.  
This will make us to do changes in both of our classes as doing arithemetic with biginteger is totally different than doing it with other data types.**

**FORM 3:- Handling larger Inputs**

import java.util.\*;

import java.io.\*;

import java.math.BigInteger;

public class MAIN\_POWER{

public static void main(String[] args) {

//creating obj of scanner class for input

Scanner sc = new Scanner(System.in);

int ex=0;

Our Big Integer variables, we don’t need to initialize them as the class itself does that

BigInteger N,PWR;

do

{

try{

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

Scanner fs = new Scanner(file);

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

while(fs.hasNext()){

Passed them to class constructor

N = fs.nextBigInteger();

PWR = fs.nextBigInteger();

POWER pw = new POWER(N,PWR);

System.out.println(N+" ^ "+PWR+" = "+pw.COMPUTE\_POWER());

}

}

catch(InputMismatchException|FileNotFoundException e){

System.out.println("Invalid input!");

}

sc.nextLine();

//asking user if he want to continue

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

ex=sc.nextInt();

}while(ex==1);

Note major changes in our POWER class

}

}

class POWER{

private static BigInteger COUNTER,NUM; //declared counter var

//constructor to initialize members

public POWER(BigInteger N,BigInteger PWR){

NUM = N;

COUNTER = PWR;

}

Assigned to the class members,

Note: you might have noticed we have not changed our class members from being static, that is because BigInteger already assigns a huge size of memory in regard to input size so by making our static we save some memory

public BigInteger COMPUTE\_POWER(){

BigInteger num=NUM; //storing the value of number

Used equals method to check if the counter is not equal to one

//loop to compute power

BigInteger.ONE , is member of BigInteger class which contain the value 1

while(!COUNTER.equals(BigInteger.ONE)){

NUM=num.multiply(NUM);

COUNTER = COUNTER.subtract(BigInteger.ONE);

**NOTE  
We cant use Increament and Decreament operators with BigIntegers , so there is subtract method for it**

}

//returning the result to main

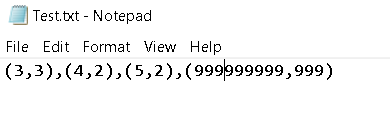
return NUM;

}

}

OUTPUT

TEXT Input file -



**NOTE - the BIGINTEGER can handle the input large enough in our drawback section, however my computer memory and processing power was not enough to handle such a large input base, further it nearly froze my computer.**

 Note – Please Zoom in to see clear output

**NOTE**

**There is still one big drawback our huge output so we format it to scientific notation by adding following code. So what we are doing is using Formatter class in text package to convert our number format into decimal scientific notation that is “0.######E0”   
The statement “**DecimalFormatSymbols.getInstance(Locale.ROOT)**”Returns an array of all locales for which the getInstance methods of this class can return localized instances.**

import java.math.BigDecimal;

Imported our classes

import java.text.NumberFormat;

import java.text.DecimalFormat;

import java.text.DecimalFormatSymbols;

Storing our huge output to string and formatting it

---- code similar ---

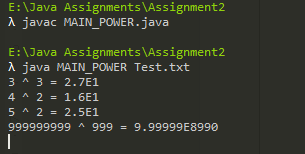
//for formatting our huge output

NumberFormat formatter = new DecimalFormat("0.######E0", DecimalFormatSymbols.getInstance(Locale.ROOT));

String str = formatter.format(pw.COMPUTE\_POWER());

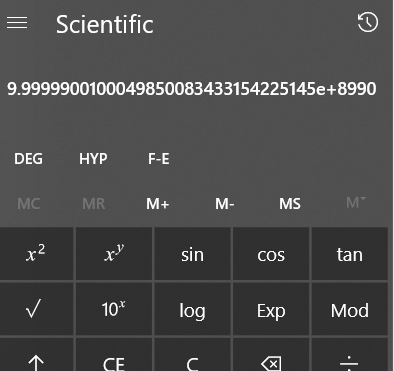
System.out.println(N+" ^ "+PWR+" = "+str);

**OUTPUT –**



We have formatted our huge output

Got this output on scientific calculator which is similar to ours how ever the formatting of decimal precision is more



# Create a class PERIMETER with the following specifications:

Private members:

LENGTH - float type

BREADTH - float type

Public members:

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

Create a parameterized constructor to accept values from actual parameters in main() to their formal parameters. Further, assign values to data members of class PERIMETER

COMPUTE\_PARAMETER() - must compute the parameter of a rectangle and return it to the main() for printing // Hint: Rectangle Parameter = 2(LENGTH + BREADTH)

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

main() - Invoke the default constructor

It must take 'length' and 'breadth' as a user input and invoke the parameterized constructor

It must then further invoke COMPUTE\_PARAMETER() and print the rectangle parameter

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

**Form1:**Acc to Que

import java.util.\*;

public class RECTANGLE{

public static void main(String[] args) {

//creating obj of scanner class for input

Scanner sc = new Scanner(System.in);

int ex=0;

float l=0,b=0;

do

{

//inputing length and breadth

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

l=sc.nextFloat();

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

b=sc.nextFloat();

PERIMETER pr = new PERIMETER(l,b);

//printing result

System.out.println(" PERIMETER :"+pr.COMPUTE\_PARAMETER()+" m");

//asking user if he want to continue

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

ex=sc.nextInt();

}while(ex==1);

}

}

class PERIMETER{

private float LENGTH,BREADTH; //declared counter var

//constructor to initialize members

public PERIMETER(float l, float b){

LENGTH = l;

BREADTH = b;

}

public float COMPUTE\_PARAMETER(){

//computing paramete

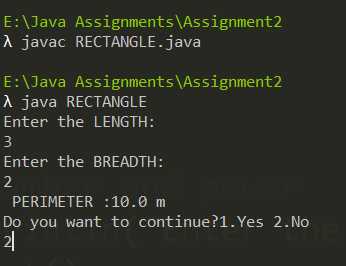
//returning the result to main

return 2\*(LENGTH+BREADTH);

}

}

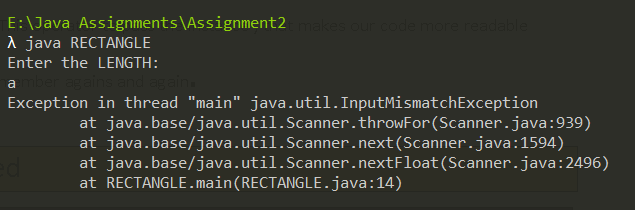
**OUTPUT:**



The Program Ask user if he wants to continue or not

**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

* **Passign many inputs through a CSV ( comma seprated file) is possible and it is good to have huge input processing in program so that user can use a data file to input**

Form 2 - Modified

import java.util.\*;

import java.io.\*;

public class RECTANGLE{

public static void main(String[] args) {

//creating obj of scanner class for input

Scanner sc = new Scanner(System.in);

int ex=0;

float l=0,b=0;

do

{

try{

//inputing number and power

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

Scanner fs = new Scanner(file);

//reading format

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

//run loop until eof(end of file)

while(fs.hasNext()){

//inputting format from file (l,b), l=length , b=breadth

l=fs.nextFloat();

b=fs.nextFloat();

//passing values of length and breadth

PERIMETER pr = new PERIMETER(l,b);

System.out.println("Length :"+l+" Breadth :"+b);

//printing result

System.out.println(" PERIMETER :"+pr.COMPUTE\_PARAMETER()+" m");

Handling every exception

}

}

catch(InputMismatchException|FileNotFoundException|ArrayIndexOutOfBoundsException e){

System.out.println("Either one of the input in file is not correct format or The file doesn't exist!!");

}

sc.nextLine(); // clearing the buffer

//asking user if he want to continue

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

ex=sc.nextInt();

}while(ex==1);

}

}

class PERIMETER{

private float LENGTH,BREADTH; //declared counter var

//constructor to initialize members

public PERIMETER(float l, float b){

LENGTH = l;

BREADTH = b;

}

public float COMPUTE\_PARAMETER(){

//computing paramete

//returning the result to main

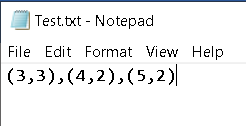
return 2\*(LENGTH+BREADTH);

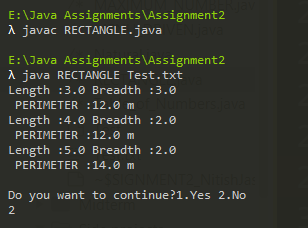
}

}

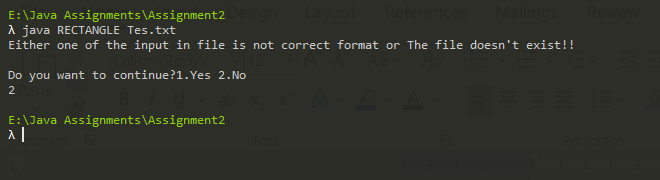
**OUTPUT**

Text File

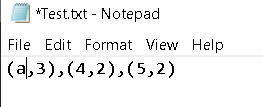




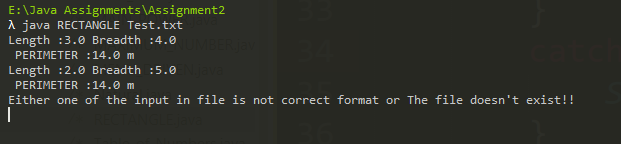
Read from file



Error : file named Tes.txt was not found



Error : for not right input type in file, we will improve this later programs that our buffer reader ignores bad input



# Create a class EVENODDPRIME with following specifications:

Private members:

NUM - integer type

Public members:

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

FINDING\_EVENODDPRIME() finds whether a number is an EVEN or ODD or PRIME or (ODD and PRIME) and return the result to main() for printing

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

main() - Invoke the default constructor

It must take an integer as user input and invoke FINDING\_EVENODDPRIME() and print the result

The program must continue as long as the user wants

FORM 1: Acc to Que

import java.util.\*;

public class Find\_Result{

public static void main(String[] args) {

//creating obj of scanner class for input

Scanner sc = new Scanner(System.in);

int N=0,ex=0;

do

{

//inputing number and power

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

N=sc.nextInt();

EVENODDPRIME eop = new EVENODDPRIME(N);

//printing result

System.out.println(N+" is "+eop.FIND\_EVENODDPRIME());

//asking user if he want to continue

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

ex=sc.nextInt();

}while(ex==1);

}

}

class EVENODDPRIME{

private int NUM; //declared counter var

//constructor to initialize members

public EVENODDPRIME(int n){

NUM = n;

}

public String FIND\_EVENODDPRIME(){

//using even and prime for deciding

boolean even=false,prime=true;

//if the number is even , even is assigned 1

if(NUM % 2 == 0){

even = true;

}

//loop for checking is prime or not

for(int i = 2; i <= NUM/2; ++i){

if(NUM % i == 0){

prime = false; //assign false to prime if number divides

break;

}

}

if(even){

if(prime){

return "even and prime"; //return string

}

else{

return "even";

}

}

else{

if(prime){

return "odd and prime";

}

else {

return "odd";

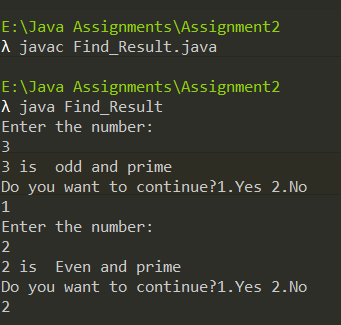
}

}

}

}

**OUTPUT:**



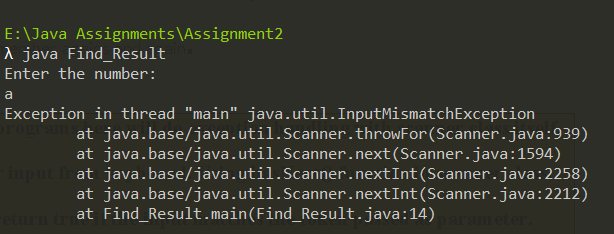
The Program Ask user if he wants to continue or not

DrawBacks -

* The algorithm can be improved further by observing that all primes are of the form 6k ± 1, with the exception of 2 and 3. This is because all integers can be expressed as (6k + i) for some integer k and for i = ?1, 0, 1, 2, 3, or 4; 2 divides (6k + 0), (6k + 2), (6k + 4); and 3 divides (6k + 3). So a more efficient method is to test if n is divisible by 2 or 3, then to check through all the numbers of form 6k ± 1

Reducing the complexity of our prime function to O(√n).

* Further it doesn’t check for case of 1 in checking for prime
* The program could use static members to improve memory usage of private members of class
* The program could be improved with exception handling Using try and catch to counter invalid user input.



* **Passign many inputs through a CSV ( comma seprated file) is possible and it is good to have huge input processing in program so that user can use a data file to input.**
* **Further, the programs nested if else is already optimal as it returns at a block where the condition become true so we don’t go through checking them all**

**At minimum we check 2 conditions , and at maximum we check 3 so that is not bad.**

FORM 2: Modified

import java.util.\*;

import java.io.\*;

public class Find\_Result{

public static void main(String[] args) {

//creating obj of scanner class for input

Scanner sc = new Scanner(System.in);

int N=0,ex=0;

do

{

try{

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

Scanner fs = new Scanner(file);

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

while(fs.hasNext()){

//inputing number from file

N=fs.nextInt();

System.out.println("the number: ");

EVENODDPRIME eop = new EVENODDPRIME(N);

//printing result

System.out.println(N+" is "+eop.FIND\_EVENODDPRIME());

}

}

catch(NoSuchElementException|FileNotFoundException|ArrayIndexOutOfBoundsException e){

System.out.println("Either one of the input in file is not correct format or The file doesn't exist!!");

}

//asking user if he want to continue

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

ex=sc.nextInt();

}while(ex==1);

}

}

class EVENODDPRIME{

private int NUM; //declared counter var

//constructor to initialize members

public EVENODDPRIME(int n){

NUM = n;

}

public String FIND\_EVENODDPRIME(){

//using even and prime for deciding

boolean even=false,prime=true;

//if the number is even , even is assigned 1

if(NUM % 2 == 0){

even = true;

}

// checking is prime or not

// Corner cases

if (NUM <= 1)

prime=false;

if (NUM <= 3)

prime=true;

// This is checked so that we can skip

// middle five numbers in below loop

if (NUM % 2 == 0 || NUM % 3 == 0)

prime=false;

for (int i = 5; i \* i <= NUM; i = i + 6)

if (NUM % i == 0 || NUM % (i + 2) == 0)

prime=false;

if(even){

if(prime){

return "even and prime"; //returns string

}

else{

return "even";

}

}

else{

if(prime){

return "odd and prime";

}

else {

return "odd";

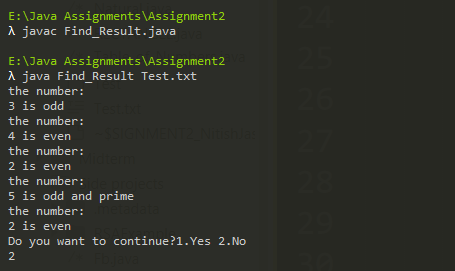
}

}

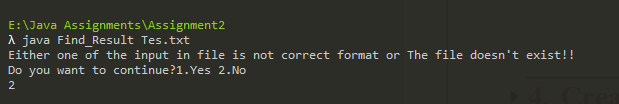
}

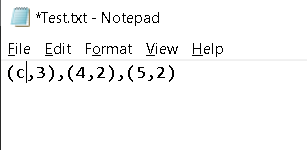
}

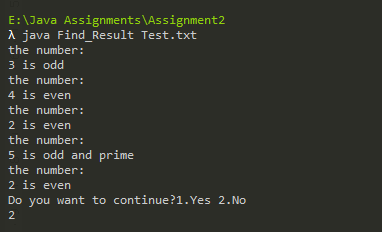
**OUTPUT:**



Read from a file







Didn’t crash and ignored the wrong format input

# Create a class ARMSTRONG\_NUMBER with following specifications:.

Private members:

NUM - integer type

Public members:

Create a default constructor to give the initial values to data members of the class ARMSTRONG\_NUMBER

COMPUTE\_ARMSTRONG() - must find whether the given number is an armstrong number or not and return the result to main() for printing

A separate class ARMSTRONG\_RESULT shall be created to include the main()

main() - Invoke the default constructor

It must take an integer as user input and invoke COMPUTE\_ARMSTRONG() and print the result

The program must continue as long as the user wants

FORM 1: acc to que

import java.util.\*;

public class Armstrong\_Result{

public static void main(String[] args) {

//creating obj of scanner class for input

Scanner sc = new Scanner(System.in);

int N=0,ex=0;

do

{

//inputing number and power

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

N=sc.nextInt();

Armstrong\_Number arm = new Armstrong\_Number(N);

//printing result

System.out.println(N + " is " + arm.Compute\_Armstrong());

//asking user if he want to continue

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

ex=sc.nextInt();

}while(ex==1);

}

}

class Armstrong\_Number{

private int NUM;

public Armstrong\_Number(int n){

NUM=n;

}

public String Compute\_Armstrong(){

//a temprory variable to store our number

//a for storing digits at diff place

//sum for storing sum of cube of each digit

int temp,a,sum=0;

temp = NUM;

for(;temp!=0;temp/=10)

{

a = temp%10;

sum +=a\*a\*a;

}

if(sum==NUM)

return " Armstrong number";

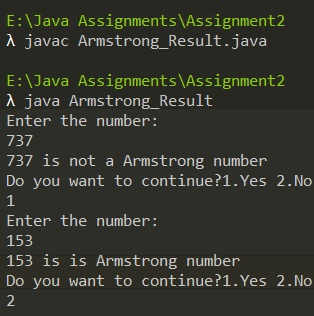
else

return "not a Armstrong number";

}

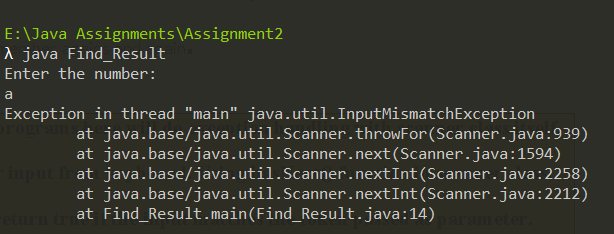
}

OUTPUT**:**



DrawBacks -

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



* **Passign many inputs through a CSV ( comma seprated file) is possible and it is good to have huge input processing in program so that user can use a data file to input.**
* **Further, the programs nested if else is already optimal as it returns at a block where the condition become true so we don’t go through checking them all**

**At minimum we check 2 conditions , and at maximum we check 3 so that is not bad**

Form 2 - Modified

import java.util.\*;

import java.io.\*;

public class Armstrong\_Result{

public static void main(String[] args) {

//creating obj of scanner class for input

Scanner sc = new Scanner(System.in);

int N=0,ex=0;

do

{

try{

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

Scanner fs = new Scanner(file);

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

while(fs.hasNext()){

//inputing number

N=fs.nextInt();

System.out.println("the number is: ");

Armstrong\_Number arm = new Armstrong\_Number(N);

//printing result

System.out.println(N + " is " + arm.Compute\_Armstrong());

}

}

catch(NoSuchElementException|FileNotFoundException|ArrayIndexOutOfBoundsException e){

System.out.println("Either one of the input in file is not correct format or The file doesn't exist!!");

}

//asking user if he want to continue

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

ex=sc.nextInt();

}while(ex==1);

}

}

class Armstrong\_Number{

private int NUM;

public Armstrong\_Number(int n){

NUM=n;

}

public String Compute\_Armstrong(){

//a temprory variable to store our number

//a for storing digits at diff place

//sum for storing sum of cube of each digit

int temp,a,sum=0;

temp = NUM;

for(;temp!=0;temp/=10)

{

a = temp%10;

sum +=a\*a\*a;

}

if(sum==NUM)

return " Armstrong number";

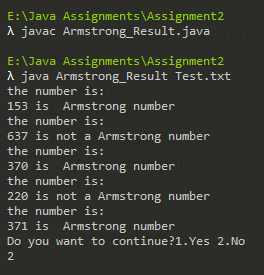
else

return "not a Armstrong number";

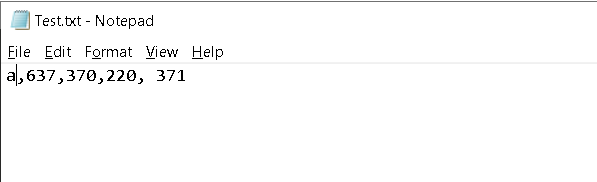
}

}

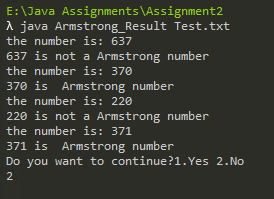
OUTPUT -



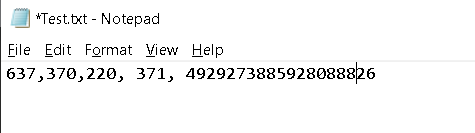
Taking input from file



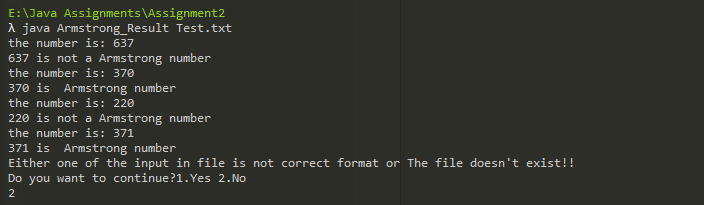
Wrong input type



Ignored the wrong input and moved to next value without showing exceptions



Large Armstrong numbers as input



Couldn’t handle large input and thrown error

# Create a class NATURAL\_NUMBERS with following specifications:

Private members:

N - integer type

Public members:

Create a default constructor to give the initial values to data members of the class NATURAL\_NUMBERS

COMPUTE\_SUMOFNATURAL() - must compute the sum of N natural numbers and return the result to main() for printing

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

main() - Invoke the default constructor

It must take an integer as user input and invoke COMPUTE\_SUMOFNATURAL() and print the result

The program must continue as long as the user wants

Form 1: acc to que

import java.util.\*;

class Natural {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int N=0,ex=0;

do

{

//inputing limit

System.out.print("Enter the limit: ");

N=sc.nextInt();

Natural\_Numbers nn = new Natural\_Numbers(N);

//printing result

System.out.println("The sum of the natural numbers till "+N+" is "+ nn.Compute\_Sumofnatural());

//asking user if he want to continue

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

ex=sc.nextInt();

}while(ex==1);

}

}

class Natural\_Numbers {

private int N;

//constructor to initialize N with the number passed in main

public Natural\_Numbers(int n){

N = n;

}

public int Compute\_Sumofnatural(){

int sum = 0;

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

{

sum += i;

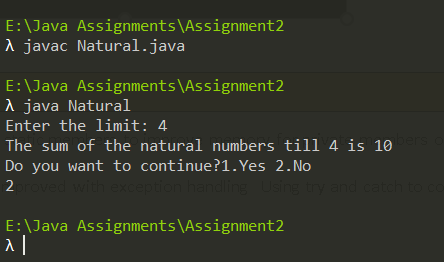
}

return sum;

}

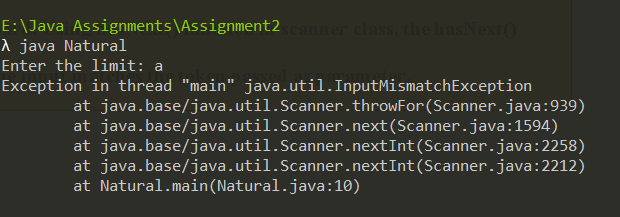
}

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.



* Now we we directly use BigInteger for our input processing.

**Note- we have also used CurrentTime method in System class to measure our execution time**

FORM 2: Modified

import java.util.\*;

import java.io.\*;

import java.math.BigInteger;

public class Natural {

public static void main(String[] args) {

//for measuring time consumed in execution

long start = System.currentTimeMillis();

Scanner sc = new Scanner(System.in);

int ex=0;

BigInteger N;

do

{

try{

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

Scanner fs = new Scanner(file);

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

while(fs.hasNext()){

//inputing limit

N=fs.nextBigInteger();

System.out.print("The limit is: "+N);

Natural\_Numbers nn = new Natural\_Numbers(N);

//printing result

System.out.println("The sum of the natural numbers till "+N+" is "+ nn.Compute\_Sumofnatural());

}

}

catch(NoSuchElementException|FileNotFoundException|ArrayIndexOutOfBoundsException e){

System.out.println("Either one of the input in file is not correct format or The file doesn't exist!!");

}

//for checkig the time consumed

long finish = System.currentTimeMillis();

System.out.println("Time consumed: " + (finish - start) + " ms");

//asking user if he want to continue

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

ex=sc.nextInt();

}while(ex==1);

}

}

class Natural\_Numbers {

private BigInteger N;

//constructor to initialize N with the number passed in main

public Natural\_Numbers(BigInteger n){

N = n;

}

public BigInteger Compute\_Sumofnatural(){

BigInteger sum = new BigInteger("0");

for(BigInteger i= N; i.compareTo(BigInteger.ZERO)>0; i=i.subtract(BigInteger.ONE))

{

BigInteger b = new BigInteger(String.valueOf(i));

sum = sum.add(b);

}

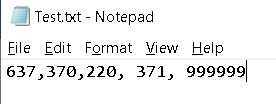
return sum;

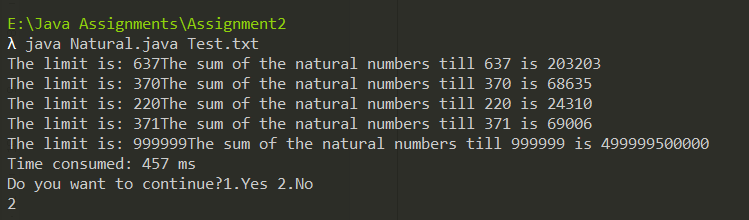
}

}

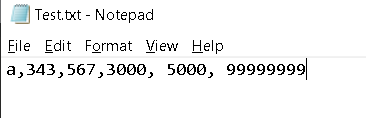
**OUTPUT:**

Text File

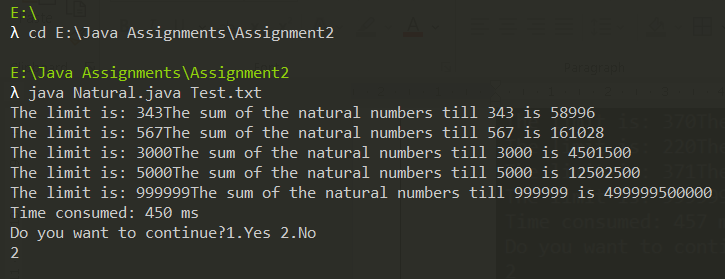




Only took 457 ms that is 0.457sec to compute such huge input



Passing in wrong input in file



Input stream igonred the bad input and moved to next value in file

# Create a class TABLE with following specifications:

Private members:

N - integer type

Public members:

Create a default constructor to give the initial values to data members of the class NATURAL\_NUMBERS

FIND\_TABLE() - must find the table of N number and print it (for example: 2\*1=2, 2\*2=4, 2\*3=6, 2\*4=8, 2\*5=10)

A separate class TABLE\_OF\_NUMBER shall be created to include the main()

main() - Invoke the default constructor

It must take an integer as user input and invoke FIND\_TABLE()

The program must continue as long as the user wants.

Form 1: acc to que

import java.util.\*;

public class Table\_of\_Numbers{

public static void main(String[] args) {

//creating scanner obj for taking inputs

Scanner sc = new Scanner(System.in);

int N=0,ex=0; //variable for inputting number and ex for asking user to continue

do

{

//inputing number

System.out.print("Enter a number : ");

N=sc.nextInt();

Table t = new Table(N);

//printing result

System.out.println("The multiplication table of "+N+" is ");

t.Find\_Table();

//asking user if he want to continue

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

ex=sc.nextInt();

}while(ex==1);

}

}

class Table {

//privatemembers

private int N;

//constructor initializing member with parameter

public Table(int n){

N = n;

}

//function to print table

public void Find\_Table(){

for(int i=1; i<=10;i++)

{

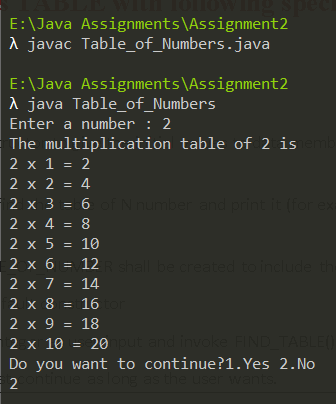
System.out.println(N + " x " + i + " = "+ N\*i);

}

}

}

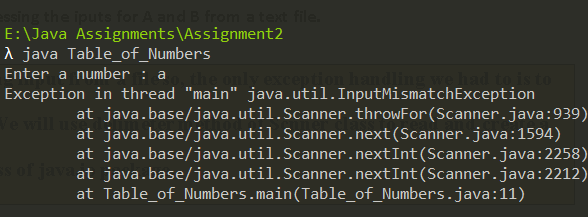
OUTPUT-



DrawBacks -

* The program could use static members to improve memory for private members of operation class
* Do exception handling for ---

“NoSuchElementException|FileNotFoundException|ArrayIndexOutOfBoundsException”



* 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.

FORM 2: Modified

import java.util.\*;

import java.io.\*;

public class Table\_of\_Numbers{

public static void main(String[] args) {

//creating scanner obj for taking inputs

Scanner sc = new Scanner(System.in);

int N=0,ex=0; //variable for inputting number and ex for asking user to continue

do

{

try{

//inputing number and power

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

Scanner fs = new Scanner(file);

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

while(fs.hasNext()){

//inputing number

N=fs.nextInt();

System.out.print("The number is: "+N);

Table t = new Table(N);

//printing result

System.out.println("The multiplication table of "+N+" is ");

t.Find\_Table();

}

}

catch(NoSuchElementException|FileNotFoundException|ArrayIndexOutOfBoundsException e){

System.out.println("Either one of the input in file is not correct format or The file doesn't exist!!");

}

//asking user if he want to continue

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

ex=sc.nextInt();

}while(ex==1);

}

}

class Table {

//privatemembers

private int N;

//constructor initializing member with parameter

public Table(int n){

N = n;

}

//function to print table

public void Find\_Table(){

for(int i=1; i<=10;i++)

{

System.out.println(N + " x " + i + " = "+ N\*i);

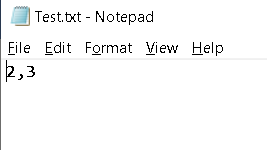
}

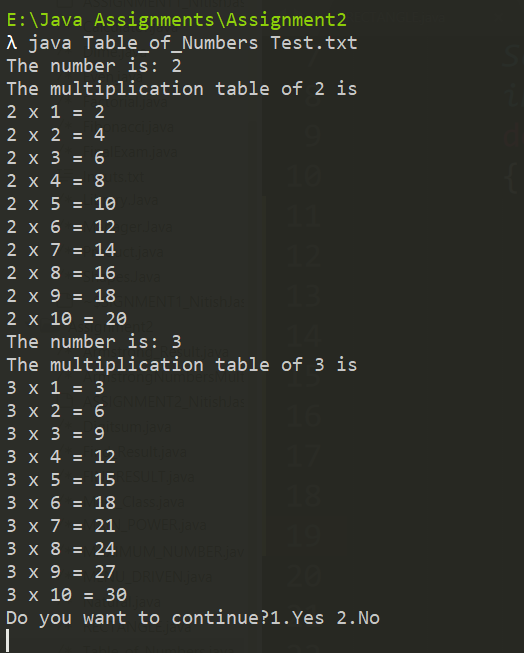
}

}

**OUTPUT:**

Our Text file for taking input -





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

# Create a class FINDING\_CHARACTER with following specifications:

Private members:

A - character type

Public members:

Create a default constructor to give the initial values to data members of the class FINDING\_CHARACTER

SEEK\_RIGHTCHARACTER() - to find whether the given user input is a capital letter, small letter, digit, or a special symbol and print it

A separate class MAIN\_CLASS shall be created to include the main()

main() - Invoke the default constructor

It must take character as a user input and invokes SEEK\_RIGHTCHARACTER()

The program must continue as long as the user wants.

FORM 1:-

import java.util.\*;

public class Main\_Class{

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

char ch = '\0';

int ex=0; //variable ex for asking user to continue

do

{

//inputing character

System.out.println("Enter a character : ");

ch = sc.next().charAt(0);

Finding\_Character fc = new Finding\_Character(ch);

//printing result

fc.Seek\_Rightcharacter();

//asking user if he want to continue

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

ex=sc.nextInt();

}while(ex==1);

}

}

class Finding\_Character {

private char A;

//constructor initializing member with parameter

public Finding\_Character(char ch) {

A = ch;

}

//function to check

public void Seek\_Rightcharacter() {

if(A>='A' && A <= 'Z')

System.out.println(A+" is a capital alphabet.");

else if(A>='a' && A <= 'z')

System.out.println(A+" is a small alphabet.");

else if(A>='0' && A<= '9')

System.out.println(A+" is a digit.");

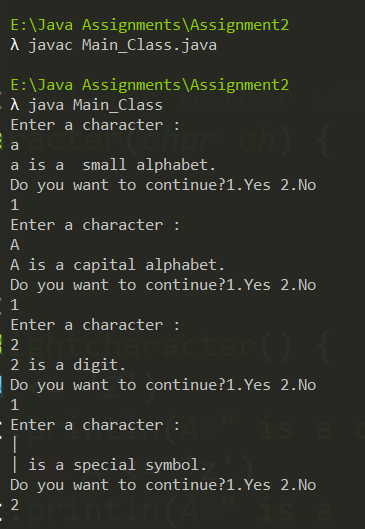
else

System.out.println(A+" is a special symbol.");

}

}

OUTPUT-



DrawBacks -

* The program could use static members to improve memory for private members of operation class
* Do exception handling for ---

“IOException|ArrayIndexOutOfBoundsException”

Note: we are not using scanner class here because in case of character we will only be able read through one leeter as our delimeter will stop at the next character cosidering as final input  
So here we use buffer reader class for file reading which contain folowing steps

* + Creation of File Descriptor for input file
  + Creation of File Reader object
  + Creation of BufferedReader object
  + Then Reading the characters from file
  + Then we organize our output with help of different lists and display it

Form 2:

import java.util.\*;

import java.io.\*;

public class Main\_Class{

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

//to maintain diff category

List <Character> Capital = new ArrayList<>();

List <Character> Small = new ArrayList<>();

List <Character> Number = new ArrayList<>();

List <Character> Special = new ArrayList<>();

char ch = '\0';

int ex=0; //variable ex for asking user to continue

do

{

try{

File f=new File("Test.txt"); //Creation of File Descriptor for input file

FileReader fr=new FileReader(f); //Creation of File Reader object

BufferedReader br=new BufferedReader(fr); //Creation of BufferedReader object

int c = 0;

while((c = br.read()) != -1) //Read char by Char

{

ch = (char) c; //converting integer to char

Finding\_Character fc = new Finding\_Character(ch);

//printing result

if(fc.Seek\_Rightcharacter()=="Capital")

Capital.add(ch);

else if(fc.Seek\_Rightcharacter()=="Small")

Small.add(ch);

else if(fc.Seek\_Rightcharacter()=="Number")

Number.add(ch);

else if(fc.Seek\_Rightcharacter()=="Special")

Special.add(ch);

}

}

catch(IOException|ArrayIndexOutOfBoundsException e){

System.out.println("Either one of the input in file is not correct format or The file doesn't exist!!");

}

System.out.println("Capital Characters : "+Capital);

System.out.println("Capital Small : "+Small);

System.out.println("Capital Number : "+Number);

System.out.println("Capital Special : "+Special);

//asking user if he want to continue

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

ex=sc.nextInt();

}while(ex==1);

}

}

class Finding\_Character {

private char A;

//constructor initializing member with parameter

public Finding\_Character(char ch) {

A = ch;

}

//function to check returns different string which are further verified in main function

public String Seek\_Rightcharacter() {

if(A>='A' && A <= 'Z')

return "Capital";

else if(A>='a' && A <= 'z')

return "Small";

else if(A>='0' && A<= '9')

return "Number";

else

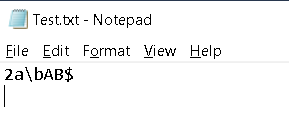
return "Special";

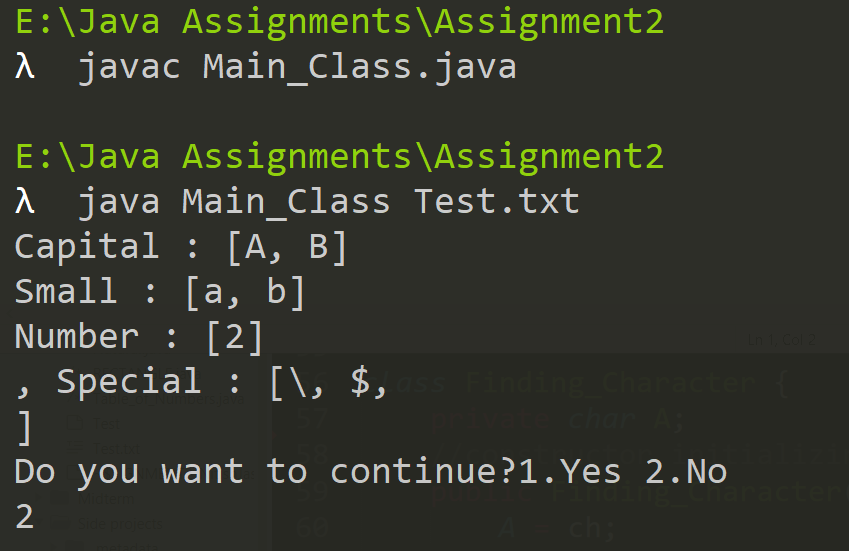
}

}

**OUTPUT:**

Our Text file for taking input -





Printed According to different categories

# Create a class DESCENDING with following specifications:

Private members:

A - integer type

B - integer type

C - integer type

Public members:

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

MAX\_SERIES() - must print the numbers in series of descending order (For example: 10 7 5) Hint: Use nested-if

A separate class MAXIMUM\_NUMBER shall be created to include the main()

main() - Invoke the default constructor

It must take 3 integer numbers as user input. As per the programming logic, you can make sure that none of the numbers entered by the user are equal and invoke MAX\_SERIES()

The program must continue as long as the user wants

FORM 1:-

import java.util.\*;

public class MAXIMUM\_NUMBER{

public static void main(String[] args) {

//creating obj of scanner class for input

Scanner sc = new Scanner(System.in);

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

do

{

//inputing number and power

System.out.println("Enter the 1st number: ");

a=sc.nextInt();

System.out.println("Enter the 1st number: ");

b=sc.nextInt();

System.out.println("Enter the 1st number: ");

c=sc.nextInt();

if(a!=b&&a!=c&&b!=c){

DESCENDING dc = new DESCENDING(a,b,c);

//printing result

System.out.println("The decending order is: ");

dc.MAX\_SERIES();

}

else{

System.out.println("The numbers are equal !!");

}

//asking user if he want to continue

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

ex=sc.nextInt();

}while(ex==1);

}

}

class DESCENDING{

private int A,B,C; //declared counter var

//constructor to initialize members

public DESCENDING(int a,int b,int c){

A = a;

B = b;

C = c;

}

public void MAX\_SERIES(){

if(A>B){

if(A>C ){

if(C>B)

System.out.println(A+" "+C+" "+B);

else

System.out.println(A+" "+B+" "+C);

}

else{

System.out.println(C+" "+A+" "+B);

}

}

else if(B>C){

if(A>C){

System.out.println(B+" "+A+" "+C);

}

else{

System.out.println(B+" "+C+" "+A);

}

}

else{

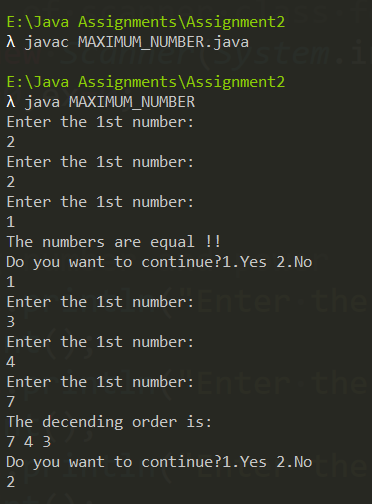
System.out.println(C+" "+B+" "+A);

}

}

}

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.
* Instead of using three variable we can use array for storing and arranging, also our if else block will also get reduced by using sorted function for data structures.

**NOTE:- we have used function clone and sort , for faster execution of our program , in out the difference with and without these functionns will be shown**

FORM 2: Modified

import java.util.\*;

import java.io.\*;

import java.math.BigInteger;

public class MAXIMUM\_NUMBER{

public static void main(String[] args) {

long start = System.currentTimeMillis();

//creating obj of scanner class for input

Scanner sc = new Scanner(System.in);

int size;

BigInteger series = new BigInteger("0");

int ex=0;

do

{

try{

//inputing number and power

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

Scanner fs = new Scanner(file);

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

while(fs.hasNext()){

//file format will be

// size,(vlaue,vlaue,......) upto size

// size,(value,value,......) upto size

size = fs.nextInt();

System.out.println("Size is:"+size);

BigInteger[] values = new BigInteger[size];

System.out.println("Series is:");

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

series = fs.nextBigInteger();

values[i] = series;

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

}

System.out.println();

DESCENDING dc = new DESCENDING(values,size);

//printing result

System.out.println("The decending order is: ");

dc.MAX\_SERIES();

System.out.println();

}

}

catch(NoSuchElementException|FileNotFoundException|ArrayIndexOutOfBoundsException e){

System.out.println("Either one of the input in file is not correct format or The file doesn't exist!!");

}

//for checkig the time consumed

long finish = System.currentTimeMillis();

System.out.println("Time consumed: " + (finish - start) + " ms");

//asking user if he want to continue

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

ex=sc.nextInt();

start=0;

finish =0 ;

}while(ex==1);

}

}

class DESCENDING{

private static int SIZE;

private static BigInteger[] SERIES = new BigInteger[SIZE];

//constructor to initialize members

public DESCENDING(BigInteger[] srs,int size){

SIZE = size;

SERIES= srs.clone(); //copying array elements to out member array

}

public void MAX\_SERIES(){

Arrays.sort(SERIES, Collections.reverseOrder());

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

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

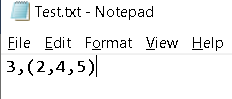
}

}

}

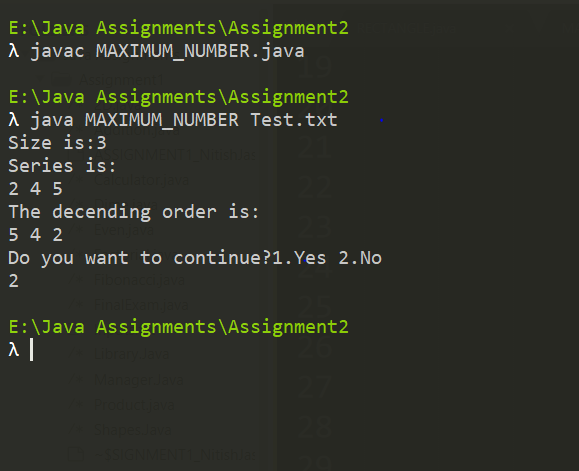
**OUTPUT:**

Our Text file for taking input -



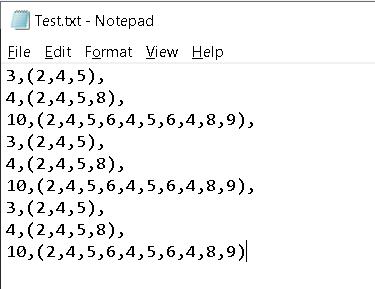
Size

Inputs



Taking series of input from the text file named “Inputs” and passing it to the array

Now checking the execution time with numerous inputs





Took us 108ms tor ead the inputs from file, pass it to array and compute the result

Now removing the **Sort** and **Clone** functions

Code in Constructor will be-

public DESCENDING(BigInteger[] srs,int size){

Removed Clone method , but left as comment for reference

SIZE = size;

//SERIES= srs.clone();

SERIES = srs;

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

SERIES[i] = srs[i];

}

}

Code in MAX\_SERIES() method –

public void MAX\_SERIES(){

//Arrays.sort(SERIES, Collections.reverseOrder());

Removed Sort method , but left as comment for reference

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

{

for (int j = i + 1; j < SIZE; j++)

{

BigInteger tmp;

int compare = SERIES[i].compareTo(SERIES[j]);

if (compare==1)

{

tmp = SERIES[i];

SERIES[i] = SERIES[j];

SERIES[j] = tmp;

}

}

}

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

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

}

}

Output – Execution time



IMPORTANT

NOTE - As you can see previous ly we got execution time of 108ms and now we got execution time of 122 ms thus using predefined functions improved the performance of our code

With difference of ------

122 – 108 = 14ms (milli seconds)

This must seems small but when the input size increases the execution time will increase however the code with predefined function will be able handle it much better than our custom code

# Create a class REVERSE\_PALINDROME with following specifications:

Private members:

NUM - integer type

Public members:

Create a default constructor to give the initial values to data members of the class REVERSE\_PALINDROME

PRINT\_REVERSE() - must compute and print the reverse of number

COMPUTE\_PALINDROME() - must find whether the given number is a plaindrome or not and then print it

A separate class MENU\_DRIVEN shall be created to include the main()

main() - Invoke the default constructor to give initial values to data members of the class REVERSE\_PALINDROME

It must take integer as a user input

Create a menu-driven program giving user a choice to either find reverse of a number or find whether number is a palindrome

It must then accordingly invoke the functions as per the choice entered by the user

The program must continue as long as the user wants

FORM 1:-

import java.util.\*;

public class MENU\_DRIVEN {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int N = 0,option=0, ex=1;

do{

//taking in the number and passing it to the constructor

//of class Reverse\_Palindrome as parameter

System.out.print("Enter any number: ");

N = sc.nextInt();

Reverse\_Palindrome rp = new Reverse\_Palindrome(N);

//switch case for menu

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

System.out.println("1.Reverse 2.Pallindrome 3.Exit");

option = sc.nextInt();

switch(option){

case 1:rp.Print\_Reverse();

break;

case 2:rp.Compute\_Palindrome();

break;

case 3:ex = 0;

break;

default: {

System.out.println("Invalid Response");

}

}

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

ex=sc.nextInt();

}while(ex==1);

}

}

class Reverse\_Palindrome {

private int NUM;

//constructor to initialize members

public Reverse\_Palindrome(int n){

NUM=n;

}

//function to print reverse of the number

public void Print\_Reverse(){

int digit=0, rev=0, Original\_NUM=0;

//storing the original number

Original\_NUM = NUM;

//take digit at 10th place

//store it in rev

// then move to digit at next place

while(NUM != 0){

digit = NUM % 10;

rev = rev \* 10 + digit;

NUM /= 10;

}

System.out.println("The reversed number is: "+ rev);

}

//function to check if the number is pallindrome

public void Compute\_Palindrome() {

int rev=0, digit=0, Original\_NUM=0;

//storing the original number

Original\_NUM = NUM;

//take digit at 10th place

//store it in rev

// then move to digit at next place

while(NUM != 0){

digit = NUM % 10;

rev = rev \* 10 + digit;

NUM /= 10;

}

//conditions to check if reverse of number is equal to original

//if it is then pallindrome , if not then its not

if(Original\_NUM == rev)

System.out.println(Original\_NUM + " is a palindrome.");

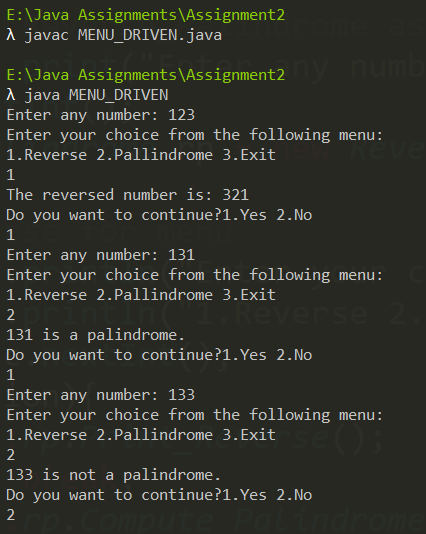
else

System.out.println(Original\_NUM + " is not a palindrome.");

}

}

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 Convert the number to string and Use predefined functions to reverse**.**

FORM 2: Modified

import java.util.\*;

import java.io.\*;

public class MENU\_DRIVEN {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int N = 0,option=0, ex=1;

do{

try{

//inputing number and power

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

Scanner fs = new Scanner(file);

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

while(fs.hasNext()){

//taking in the number and passing it to the constructor

//of class Reverse\_Palindrome as parameter

N = fs.nextInt();

Reverse\_Palindrome rp = new Reverse\_Palindrome(N);

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

//switch case for menu

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

System.out.println("1.Reverse 2.Pallindrome 3.Exit");

option = sc.nextInt();

switch(option){

case 1:rp.Print\_Reverse();

break;

case 2:rp.Compute\_Palindrome();

break;

case 3:ex = 0;

System.exit(0);

default: {

System.out.println("Invalid Response");

}

}

}

}

catch(NoSuchElementException|FileNotFoundException|ArrayIndexOutOfBoundsException e){

System.out.println("Either one of the input in file is not correct format or The file doesn't exist!!");

}

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

ex=sc.nextInt();

}while(ex==1);

}

}

class Reverse\_Palindrome {

private int NUM;

//constructor to initialize members

public Reverse\_Palindrome(int n){

NUM=n;

}

//function to print reverse of the number

public void Print\_Reverse(){

int digit=0, rev=0, Original\_NUM=0;

//storing the original number

Original\_NUM = NUM;

//take digit at 10th place

//store it in rev

// then move to digit at next place

while(NUM != 0){

digit = NUM % 10;

rev = rev \* 10 + digit;

NUM /= 10;

}

System.out.println("The reversed number is: "+ rev);

}

//function to check if the number is pallindrome

public void Compute\_Palindrome() {

int rev=0, digit=0, Original\_NUM=0;

//storing the original number

Original\_NUM = NUM;

//take digit at 10th place

//store it in rev

// then move to digit at next place

while(NUM != 0){

digit = NUM % 10;

rev = rev \* 10 + digit;

NUM /= 10;

}

//conditions to check if reverse of number is equal to original

//if it is then pallindrome , if not then its not

if(Original\_NUM == rev)

System.out.println(Original\_NUM + " is a palindrome.");

else

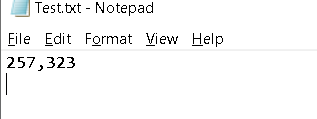
System.out.println(Original\_NUM + " is not a palindrome.");

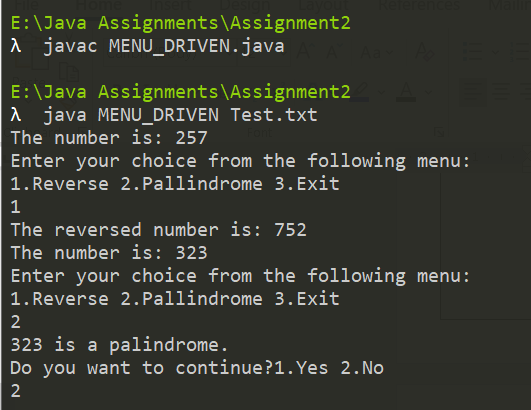
}

}

**OUTPUT:**

Our Text file for taking input -





Gone to next input

Taking input from file

# Q10. Create a class SUMOFDIGITS with following specifications:

Private members:

NUM - integer type

Public members:

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

COMPUTE\_SUMOFDIGITS() - must compute the sum of digits and return it to main() for printing

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

main() - Invoke the default constructor

It must take an integer as a user input and invoke COMPUTE\_SUMOFDIGITS()

The program must continue as long as the user wants

FORM 1:- Acc to the question

import java.util.\*;

public class Digitsum {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int N=0,ex=0;

do

{

//inputing number

System.out.println("Enter a number : ");

N = sc.nextInt();

Sumofdigits sd = new Sumofdigits(N);

//printing result

System.out.println("The sum of the digits of "+N+" is : "+sd.Compute\_Sumofdigits());

//asking user if he want to continue

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

ex=sc.nextInt();

}while(ex==1);

}

}

class Sumofdigits {

private int NUM;

//constructor to initialize members

public Sumofdigits(int n) {

NUM = n;

}

public int Compute\_Sumofdigits(){

int digit=0, sum=0;

//getting digit at 10s place

//adding it to sum

//moving to digit at other places

while(NUM>0){

digit = NUM % 10;

sum += digit;

NUM/= 10;

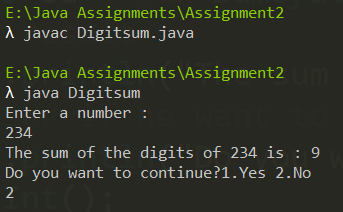
}

return sum;

}

}

OUTPUT-



DrawBacks -

* The program could use static members to improve memory for private members of operation class
* Do exception handling for “**NoSuchElementException|FileNotFoundException|ArrayIndexOutOfBoundsException**”
* Program is not made for taking series of inputs, so like from previous example, we take input form a text file**.**

FORM 2: Modified

import java.util.\*;

import java.io.\*;

import java.math.BigInteger;

public class Digitsum {

public static void main(String[] args) {

long start = System.currentTimeMillis();

Scanner sc = new Scanner(System.in);

BigInteger N;

int ex=0;

do

{

try{

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

Scanner fs = new Scanner(file);

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

while(fs.hasNext()){

//inputing number

N = fs.nextBigInteger();

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

Sumofdigits sd = new Sumofdigits(N);

//printing result

System.out.println("The sum of the digits of "+N+" is : "+sd.Compute\_Sumofdigits());

//asking user if he want to continue

}

}

catch(NoSuchElementException|FileNotFoundException|ArrayIndexOutOfBoundsException e){

System.out.println("Either one of the input in file is not correct format or The file doesn't exist!!");

}

//for checkig the time consumed

long finish = System.currentTimeMillis();

System.out.println("Time consumed: " + (finish - start) + " ms");

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

ex=sc.nextInt();

}while(ex==1);

}

}

class Sumofdigits {

private static BigInteger NUM;

//constructor to initialize members

public Sumofdigits(BigInteger n) {

NUM = n;

}

public BigInteger Compute\_Sumofdigits(){

BigInteger digit= new BigInteger("0");

BigInteger sum= new BigInteger("0");

//getting digit at 10s place

//adding it to sum

//moving to digit at other places

while(!NUM.equals(BigInteger.ZERO)){

digit = NUM.mod(new BigInteger("10"));

sum = sum.add(digit);

NUM= NUM.divide(new BigInteger("10"));

}

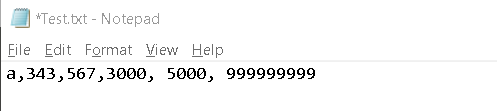
return sum;

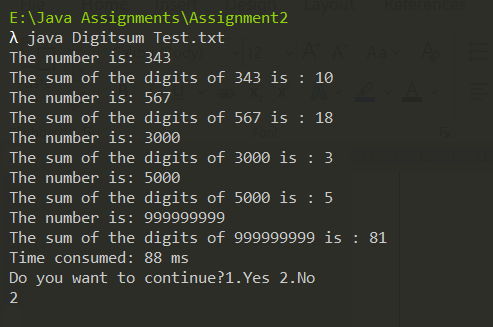
}

}

**OUTPUT:**

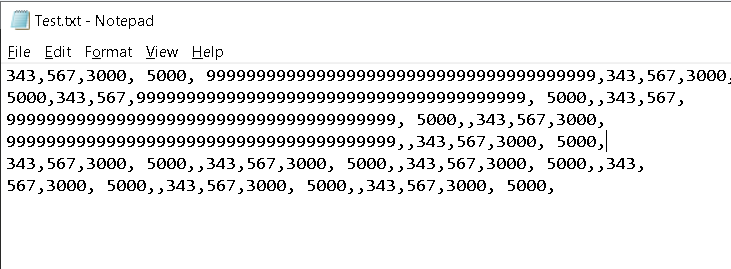
Our Text file for taking input -



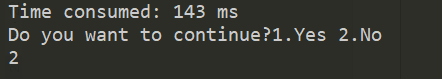


Took 88ms for 5 input and one input with 9 unit places

Ignored the first bad input



Increasing our input size and checking our time of execution



Gave us 143 ms

The difference is not that much

143 – 88 = 55 ms

If we comapre it with the input size which was only 5 inputs previously our program did great in terms of executiontime

# References

* Oracle.(2020). Class BigInteger . Retrived from <https://docs.oracle.com/javase/7/docs/api/java/math/BigInteger.html> (30th Oct, 2020)
* Oracle.(2020). Class NumberFormat . Retrived from <https://docs.oracle.com/javase/7/docs/api/java/text/NumberFormat.html>

(28th Oct, 2020)

* Oracle.(2020). Class DecimalFormatSymbols . Retrived from <https://docs.oracle.com/javase/8/docs/api/java/text/DecimalFormatSymbols.html>

(28th Oct, 2020)

* StackOverflow.(2015,Feb 1). How to 'pre increment' a BigInteger in Java?. Retrieved from <https://stackoverflow.com/questions/28264656/how-to-pre-increment-a-biginteger-in-java> (29th Oct,2020)
* Sonarsource.(2020).Java Rules.Rules for using operator with Big Integer.Retrieved from <https://rules.sonarsource.com/java/RSPEC-881>(30th Oct, 2020)