| NAME | JACOB JOHN |
|---|---|
| REGISTER NO. | 16BCE2205 |
| E-MAIL | jacob.john2016@vitstudent.ac.in |
| COURSE | Java Programming |


## *CAT 2*

**You are required to compute the power of a number by implementing a calculator. Create a class MyCalculator which consists of a single method long power(int, int). This method takes two integers, n and p, as parameters and finds $n^p$. If either n or p is negative, then the method must throw an exception which says "n and p should not be negative". Also, if both n and p are zero, then the method must throw an exception which says " n and p should not be zero "**

**Complete the function power in class MyCalculator and return the appropriate result after the power operation or an appropriate exception as detailed above.**

### Code

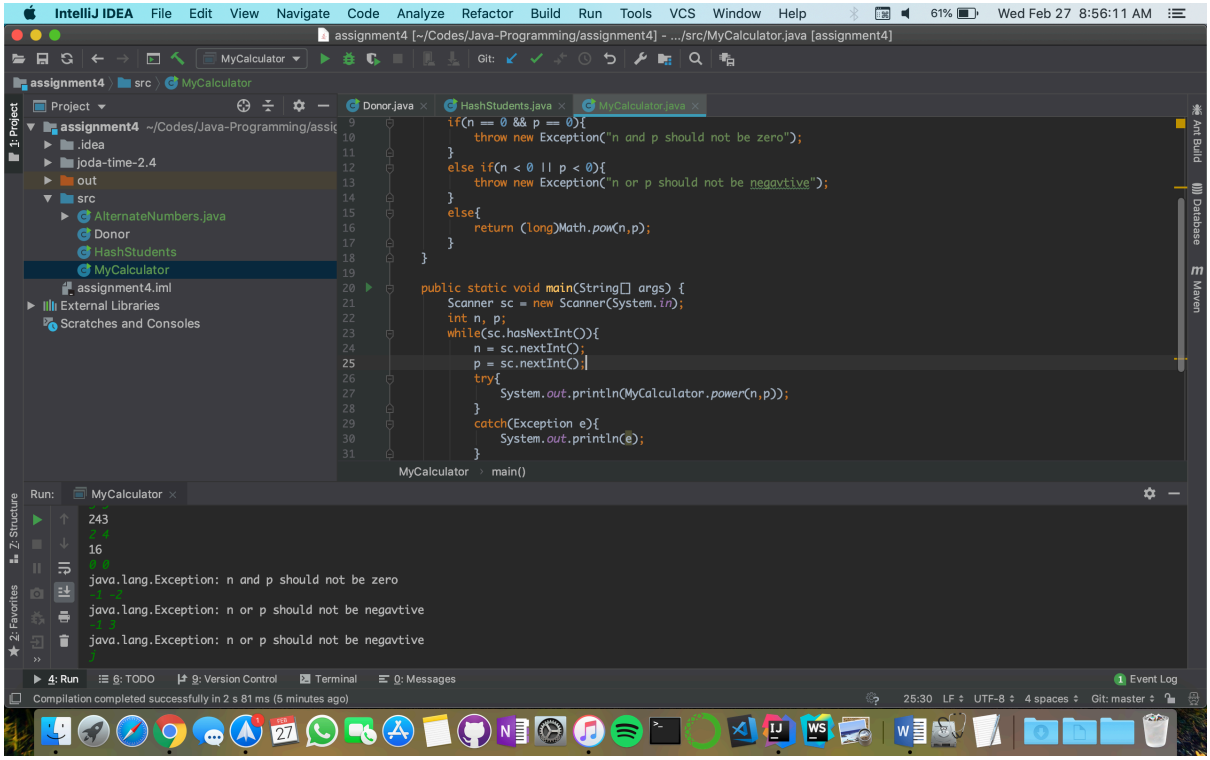
```
/*
 * Complete the function power in class MyCalculator and return the appropriate
result
 * after the power operation or an appropriate exception as detailed above.
 */
import java.util.Scanner;

public class MyCalculator{
    static long power(int n, int p) throws Exception{
        if(n == 0 && p == 0){
            throw new Exception("n and p should not be zero");
        }
        else if(n < 0 || p < 0){
            throw new Exception("n or p should not be negavtive");
        }
        else{
            return (long)Math.pow(n,p);
        }
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n, p;
        while(sc.hasNextInt()){
            n = sc.nextInt();
            p = sc.nextInt();
            try{
                System.out.println(MyCalculator.power(n,p));
            }
            catch(Exception e){
                System.out.println(e);
            }
        }
```

```
    }
}
```

## Output


IntelliJ IDEA File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
assignment4 [~/Codes/Java-Programming/assignment4] - .../src/MyCalculator.java [assignment4]
Donor.java
HashStudents.java
MyCalculator.java
if(n == 0 && p == 0){
throw new Exception("n and p should not be zero");
}
else if(n < 0 || p < 0){
throw new Exception("n or p should not be negavtive");
}
else{
return (long)Math.pow(n,p);
}
}
public static void main(String[] args) {
Scanner sc = new Scanner(System.in);
int n, p;
while(sc.hasNextInt()){
n = sc.nextInt();
p = sc.nextInt();
try{
System.out.println(MyCalculator.power(n,p));
}
catch(Exception e){
System.out.println(e);
}
Run: MyCalculator
243
2 4
16
0 0
java.lang.Exception: n and p should not be zero
-1 -4
java.lang.Exception: n or p should not be negavtive
-1 3
java.lang.Exception: n or p should not be negavtive
1
Compilation completed successfully in 2 s 81 ms (5 minutes ago)

**Write a Java program to Print alternate numbers using 2 Threads.**

## Code

```
// Write a Java program to Print alternate numbers using 2 Threads.

class Numbers{
    int n = 0;
    int end;
    Numbers(int end){
        this.end = end;
    }

    synchronized void printEvenNum(){
        while(n<=end){
            if(n%2 == 0){
                System.out.println(Thread.currentThread().getName() + " is
printing " + n);
                n++;
                notify();
            }
            else if(n%2 != 0){
                try {
                    wait();
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
        }
    }

    synchronized void printOddNum(){
        while(n<=end){
            if(n%2 != 0){
                System.out.println(Thread.currentThread().getName() + " is
printing " + n);
                n++;
                notify();
            }
            else if(n%2 == 0){
                try {
                    wait();
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
        }
    }
}

public class AlternateNumbers {
    public static void main(String[] args) {
        final Numbers num = new Numbers(100);

        new Thread(){
            public void run(){
                num.printEvenNum();
            }
```

```
        }.start();
        new Thread(){
            public void run(){
                num.printOddNum();
            }
        }.start();
    }
}
```

## Output