Started on	Thursday, 24 August 2023, 12:15 PM
State	Finished
Completed on	Thursday, 24 August 2023, 1:02 PM
Time taken	46 mins 37 secs
Name	BALAJI S CSD

Question **1**

Correct

Marked out of 1.00

Write a program that accepts 5 inputs and returns the count of how many of those 5 are odd.

For example,

If the five inputs are 12, 17, 19, 14, and 115, there are three odd numbers 17, 19 and 115. So, the program must return 3.

Similarly,

If the five inputs are 15, 0, -12, 19, and 28, there are two odd numbers 15 and 19. So, the program must return 2.

Observe that zero is considered an even number.

For example:

Input	Result
12 17 19 14 115	3
15 0 -12 19 28	2

Answer: (penalty regime: 0 %)

```java
import java.util.*;
public class Odd
{
    public static void main(String args[])
    {
        Scanner scan=new Scanner(System.in);
        int x=0;
        int no ;
        int i ;
        {
            for(i=1;i<=5;i++)
            {
                no= scan.nextInt();
                if (no%2!=0)
                x++;

            }
            System.out.println(x);
        }
    }
}
```

	Input	Expected	Got	
✓	12 17 19 14 115	3	3	✓
✓	15 0 -12 19 28	2	2	✓

Passed all tests! ✓

Question **2**

Correct

Marked out of 1.00

In mathematics, the factorial of a non-negative integer n, denoted by n!, is the product of all positive integers less than or equal to n. For example,

5! = 5 x 4 x 3 x 2 x 1 = 120

4! = 4 x 3 x 2 x 1 = 24

9! = 9 x 8 x 7 x 6 x 5 x 4 x 3 x 2 x 1 = 362880

Write a program to find the factorial of a given number.

The given number will be passed to the program as an input of type int.

The program is expected to calculate the factorial of the given number and return it as an int type.

Assumptions for this program:

The given input number will always be greater than or equal to 1.

Due to the range supported by int. the input numbers will range from 1 to 12.

For example:

Input	Result
5	120
4	24
9	362880

Answer: (penalty regime: 0 %)

```
import java.util.*;
public class Gold
{
    public static void main(String args[])
    {
        Scanner scanner=new Scanner(System.in);
        int x =scanner.nextInt();
        int fact=x;
        for(int i=1;i<x;i++)
        {
            fact=fact*i;
        }
        System.out.println(fact);
    }
}
```

	Input	Expected	Got	
✓	5	120	120	✓
✓	4	24	24	✓
✓	9	362880	362880	✓

Passed all tests! ✓

Question **3**

Correct

Marked out of 1.00

Determine the factors of a number (i.e., all positive integer values that evenly divide into a number) and then return the pth element of the list, sorted ascending. If there is no pth element, return 0.

Example

n = 20

p = 3

The factors of 20 in ascending order are {1, 2, 4, 5, 10, 20}. Using 1-based indexing, if p = 3, then 4 is returned. If p > 6, 0 would be returned.

Constraints

$1 \le n \le 10^{15}$

$1 \le p \le 10^{9}$

The first line contains an integer n, the number to factor.

The second line contains an integer p, the 1-based index of the factor to return.

Sample Case 0

Sample Input 0

10

3

Sample Output 0

5

Explanation 0

Factoring n = 10 results in {1, 2, 5, 10}. Return the p = 3rd factor, 5, as the answer.

Sample Case 1

Sample Input 1

10

5

Sample Output 1

0

Explanation 1

Factoring n = 10 results in {1, 2, 5, 10}. There are only 4 factors and p = 5, therefore 0 is returned as the answer.

Sample Case 2

Sample Input 2

1

1

Sample Output 2

1

Explanation 2

Factoring n = 1 results in {1}. The p = 1st factor of 1 is returned as the answer.

For example:

Input	Result
10 3	5
10 5	0

Input	Result
1 1	1

Answer: (penalty regime: 0 %)

```java
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
import java.util.Scanner;
public class Main
{
    public static void main (String[]args)
    {
        Scanner scanner=new Scanner(System.in);
        int n=scanner.nextInt();
        int p=scanner.nextInt();
        List<Integer>factors= new ArrayList<>();
        for(int i=1;i<=n;i++)
        {
            if(n%i==0)
              factors.add(i);

        }
        Collections.sort(factors);
        if (p>0 && p<=factors.size())
        {
            int result=factors.get(p-1);
            System.out.println(result);

        }
        else
        {
            System.out.println(0);
        }

    }
}
```

	Input	Expected	Got	
✓	10 3	5	5	✓
✓	10 5	0	0	✓
✓	1 1	1	1	✓

Passed all tests! ✓

Question **4**

Correct

Marked out of 1.00

Rakesh loves playing with numbers. He took the Fibonacci series and wants to find the sum of squares of the series until a given value. Write a code that implements his task.

Input Format:

Single Integer N

Output Format:

Display the sum of squares of the Fibonacci series until the Nth term.

Example Input:

9

Output:

1870

Explanation:

The numbers are: 1 1 2 3 5 8 13 21 34

Sum of their squares is: 1 + 1 + 4 + 9 + 25 + 64 + 169 + 441 + 1156 = 1870

For example:

Input	Result
9	1870

Answer: (penalty regime: 0 %)

```java
import java.util.*;
public class Balu
{
    public static void main(String[]args)
    {
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        int a=0;int b=1;int c;
        int sum =0;
        c=a+b;
        for(int i=0;i<n;i++)
        {
            a=b;
            b=c;
            c=a+b;
            sum=sum+a*a;

        }
        System.out.println(sum);
    }
}
```

	Input	Expected	Got	
✓	9	1870	1870	✓

Passed all tests! ✓

Question **5**

Correct

Marked out of 1.00

Write a program that finds and returns the Nth prime number. N will be passed as input to the program.

Assumption: 1 <= N <=1000, where N is the position of the prime number

The first prime number is 2

The second prime number is 3

The third prime number is 5

The fourth prime number is 7

The fifth prime number is 11

. . . and so on.

Example1: If the given number N is 10, the method must return the 10th prime number i.e. 29

Example2: If the given number N is 13, the method must return the 13th prime number i.e. 41

For example:

Input	Result
10	29
13	41

Answer: (penalty regime: 0 %)

```
import java.util.*;
public class Mass
{
    public static void main(String[]args)
    {
        Scanner scanner= new Scanner(System.in);
        int n=scanner.nextInt();
        int count=0;
        int num=1,i;
        while(count<n)
        {
            num= num+1;
            for(i=2;i<=num;i++)
            {
                if(num%i==0)
                 break;

            }
            if(i==num)
             count=count+1;
        }
        System.out.println(num);
    }
}
```

	Input	Expected	Got	
✓	10	29	29	✓
✓	13	41	41	✓

Passed all tests! ✓