Introduction to Java

Milestone 1

Table of Contents

[Lecture 1: Flowcharts 3](#_Toc135918111)

[Average of three numbers 3](#_Toc135918112)

[Check Number 4](#_Toc135918113)

[Valid Triangle 4](#_Toc135918114)

[Find Product 4](#_Toc135918115)

[Print Even Numbers 5](#_Toc135918116)

[Assignment 5](#_Toc135918117)

[Check triangle 5](#_Toc135918118)

[Sum of evens 5](#_Toc135918119)

[Find GCD 6](#_Toc135918120)

[Check Prime number 6](#_Toc135918121)

[All Fibonacci numbers 6](#_Toc135918122)

[Member of Fibonacci 7](#_Toc135918123)

[Lecture 2: Getting Started with Java 8](#_Toc135918124)

[Size of long 8](#_Toc135918125)

[Datatype 8](#_Toc135918126)

[Boolean Variable 8](#_Toc135918127)

[Valid Variable Name 8](#_Toc135918128)

[Uninitialized Local Variable 8](#_Toc135918129)

[Predict the output 8](#_Toc135918130)

[Predict The String output 9](#_Toc135918131)

[Multiple inputs 9](#_Toc135918132)

[Integer and String 9](#_Toc135918133)

[Average Marks 10](#_Toc135918134)

[Lecture 3 : How is Data Stored? 10](#_Toc135918135)

[Short data type 10](#_Toc135918136)

[Byte 10](#_Toc135918137)

[Check for Error 11](#_Toc135918138)

[Figure out the Output 11](#_Toc135918139)

[Figure out the Output 11](#_Toc135918140)

[Automatic type conversion 11](#_Toc135918141)

[Lecture 4 : Conditionals and Loops 12](#_Toc135918142)

[Error in code 12](#_Toc135918143)

[Find the output 12](#_Toc135918144)

[Predict the output 12](#_Toc135918145)

[Predict the output 12](#_Toc135918146)

[Find output 13](#_Toc135918147)

[Find output 13](#_Toc135918148)

[Find the output 14](#_Toc135918149)

[Figure out the output 14](#_Toc135918150)

[What will be the output 14](#_Toc135918151)

[Positive Negative 15](#_Toc135918152)

[Find the output 15](#_Toc135918153)

[Find the output 16](#_Toc135918154)

[Find Character Case 16](#_Toc135918155)

[Find the output 17](#_Toc135918156)

[Number of hello 17](#_Toc135918157)

[Number of hello 2 18](#_Toc135918158)

[Same output 18](#_Toc135918159)

[Fahrenheit to Celsius Table 19](#_Toc135918160)

[Primality Check 20](#_Toc135918161)

[Find the output 21](#_Toc135918162)

[Check the error 21](#_Toc135918163)

[Check the error 22](#_Toc135918164)

[Check the error 22](#_Toc135918165)

[Check the error 22](#_Toc135918166)

[What is the output 23](#_Toc135918167)

[Assignment 23](#_Toc135918168)

[Total Salary 23](#_Toc135918169)

[Multiplication Table 24](#_Toc135918170)

[Sum of even & odd 25](#_Toc135918171)

[Factors 26](#_Toc135918172)

[Find power of a number 27](#_Toc135918173)

[Lecture 5 : Patterns 1 28](#_Toc135918174)

[Code : Square Pattern 28](#_Toc135918175)

[Code : Triangular Star Pattern 29](#_Toc135918176)

[Code : Triangle Number Pattern 30](#_Toc135918177)

[Code : Reverse Number Pattern 31](#_Toc135918178)

[Code : Alpha Pattern 32](#_Toc135918179)

[Code : Character Pattern 33](#_Toc135918180)

[Code : Interesting Alphabets 34](#_Toc135918181)

[Lecture 6 : Patterns 2 35](#_Toc135918182)

[Code : Mirror Image Number Pattern 35](#_Toc135918183)

[Code : Inverted Number Pattern 36](#_Toc135918184)

[Code : Star Pattern 37](#_Toc135918185)

[Code : Triangle of Numbers 37](#_Toc135918186)

[Code : Diamond of stars 39](#_Toc135918187)

[Assignment 40](#_Toc135918188)

[Half Diamond Pattern 40](#_Toc135918189)

[Parallelogram Pattern 41](#_Toc135918190)

[Sum Pattern 42](#_Toc135918191)

[Odd Square 43](#_Toc135918192)

# Lecture 1: Flowcharts

## Average of three numbers

You are given three numbers. You need to calculate and print their average value. Draw a flowchart for this process.

Flowchart: -

Start

Input three numbers (num1, num2, num3)

Calculate the sum of the three numbers (sum = num1 + num2 + num3)

Calculate the average by dividing the sum by 3 (average = sum / 3)

Print the average value

Stop

End

## Check Number

You are given a single number. You need to print one of the following outputs according to the number's nature.

Print 1, if the number is positive

Print -1, if it's negative

Print 0, if it's equal to 0

Draw a flowchart for this process.

Flowchart: -

Start

Input a number (num)

If num > 0, then

Print "1" (Positive number)

If num < 0, then

Print "-1" (Negative number)

If num = 0, then

Print "0" (Equal to zero)

Stop

End

## Valid Triangle

You are given 3 numbers. Each number represents the length of a line. You need to figure out whether these lines can form a valid triangle.

If a valid triangle can be formed, print "Yes", otherwise print "No".

Draw a flowchart for this process

A triangle is a **valid triangle**, If and only If, the sum of any two sides of a triangle is greater than the third side. For Example, let A, B and C are three sides of a triangle. Then, A + B > C, B + C > A and C + A > B

Flowchart:

Start

Read the lengths of the three lines (a, b, c)

Check if a + b > c and a + c > b and b + c > a

If the condition is true, go to step 5; otherwise, go to step 6

Print "Yes" (A valid triangle can be formed)

Print "No" (A valid triangle cannot be formed)

Stop

## Find Product

You are given a single non-negative integer, N. You need to calculate and print N factorial (N!)

N factorial is defined as the product of all integers from 1 to N (both inclusive)

Draw a flowchart for this process.

Flowchart:

Start

Read the value of N

Set the initial value of the factorial (fact) to 1

Set the initial value of the counter (i) to 1

Check if i is less than or equal to N

If the condition is true, go to step 7; otherwise, go to step 10

Multiply fact by i and store the result in fact

Increment the value of i by 1

Go back to step 5

Print the value of fact as the factorial of N

Stop

## Print Even Numbers

You are given a single positive integer, N. You need to print all even integers that occur between 1 and N (both inclusive).

Draw a flowchart for this process

Flowchart:

Start

Read the value of N

Set the initial value of the counter (num) to 1

Check if num is less than or equal to N

If the condition is true, go to step 6; otherwise, go to step 9

Check if num is even

If the condition is true, print num

Increment the value of num by 1

Go back to step 4

Stop

# Assignment

## Check triangle

You are given the lengths of 3 sides of a valid triangle. You need to print any one of the following outputs depending on the triangle's nature.

Print 1, if the triangle is equilateral

Print 0, if it's isosceles

Print -1, if it's scalene

Draw a flowchart for this process.

Flowchart:

Start

Read the lengths of the three sides of the triangle (a, b, c)

Check if a equals b and b equals c

If the condition is true, print 1 (Equilateral triangle) and go to step 9

Check if a equals b or b equals c or a equals c

If the condition is true, print 0 (Isosceles triangle) and go to step 9

Print -1 (Scalene triangle)

Go to step 9

Stop

## Sum of evens

You are given a single positive integer, N. You need to calculate and print the sum of all even numbers till N(inclusive)

Draw a flowchart for this process.

Flowchart:

Start

Read the value of N

Set the initial value of the sum (total) to 0

Set the initial value of the counter (num) to 2

Check if num is less than or equal to N

If the condition is true, go to step 7; otherwise, go to step 11

Check if num is even

If the condition is true, add num to total and update the value of total

Increment the value of num by 2 (to skip odd numbers)

Go back to step 5

Print the value of total as the sum of even numbers up to N

Stop

## Find GCD

You are given two numbers. You need to calculate and print their greatest common divisor (GCD).

Draw a flowchart for this process.

Flowchart:

Start

Read the values of the two numbers (num1, num2)

Set the initial value of the GCD variable (gcd) to 1

Set the initial value of the divisor (div) to 1

Check if div is less than or equal to num1 and div is less than or equal to num2

If the condition is true, go to step 7; otherwise, go to step 12

Check if num1 is divisible by div and num2 is divisible by div

If the condition is true, update the value of gcd to div

Increment the value of div by 1

Go back to step 5

Print the value of gcd as the greatest common divisor of num1 and num2

Stop

## Check Prime number

You are given a single positive integer, N. You need to find and print whether N is Prime or not.

Draw a flowchart for this process.

Flowchart:

Start

Read the value of N

Set the initial value of the flag variable (isPrime) to true

Set the initial value of the divisor (div) to 2

Check if div is less than N

If the condition is true, go to step 7; otherwise, go to step 11

Check if N is divisible by div

If the condition is true, set isPrime to false and go to step 11

Increment the value of div by 1

Go back to step 5

Check the value of isPrime

If isPrime is true, print "Prime"

If isPrime is false, print "Not Prime"

Stop

## All Fibonacci numbers

You are given a single non-negative integer, N. You need to print all numbers that:

(i) occur in the range 0 to N (both inclusive)

(ii) are a part of the fibonacci sequence

Draw a flowchart for this process.

Flowchart:

Start

Read the value of N

Set the initial values of the variables:

fib1 = 0 (the first number in the Fibonacci sequence)

fib2 = 1 (the second number in the Fibonacci sequence)

fib3 = 0 (initially set to 0)

Print fib1 (as it is the first number in the Fibonacci sequence)

Print fib2 (as it is the second number in the Fibonacci sequence)

Set the initial value of the counter (i) to 3

Check if fib3 is less than or equal to N

If the condition is true, go to step 9; otherwise, go to step 13

Calculate the next Fibonacci number:

fib3 = fib1 + fib2

Update the values of fib1 and fib2 accordingly

Check if fib3 is less than or equal to N

If the condition is true, print fib3

Increment the value of i by 1 and go back to step 7

Stop

## Member of Fibonacci

You are given a single non-negative integer, N. You need to find out whether N is a part of the fibonacci sequence.

Print "Yes" if it is and "No" if it's not.

Draw a flowchart for this process.

Flowchart:

Start

Read the value of N

Set the initial values of the variables:

fib1 = 0 (the first number in the Fibonacci sequence)

fib2 = 1 (the second number in the Fibonacci sequence)

isFibonacci = false (initially set to false)

Check if N equals fib1 or N equals fib2

If the condition is true, set isFibonacci to true and go to step 9

Set the initial value of the counter (i) to 3

Calculate the next Fibonacci number:

fib3 = fib1 + fib2

Update the values of fib1 and fib2 accordingly

Check if N equals fib3

If the condition is true, set isFibonacci to true

Print "Yes" if isFibonacci is true; otherwise, print "No"

Stop

# Lecture 2: Getting Started with Java

## Size of long

Primitive data type 'long' is having size \_\_\_\_\_\_\_ byte(s) in Java Programming.

Answer:

8

## Datatype

Which of the following data type stores longest decimal number?

Correct Answer

double

Solution Description

Out of all given options, only float and double can hold decimal numbers. Size of the float is 4 bytes and double are 8 bytes (in most of the compilers, as the size of data types is compiler specific). So double can store bigger decimal numbers.

## Boolean Variable

Which of these values can be assigned to a boolean variable in Java?

Correct Answer

c. true or false

Solution Description

Boolean variable in java can only hold either true or false value.

## Valid Variable Name

Which of these is a valid variable name?

Correct Answer

Var1

Solution Description

In java, you cannot start your variable name with numbers and, they cannot contain any other special character except underscore (\_) and dollar ($).

## Uninitialized Local Variable

Compiler never assigns a default value to an uninitialized local variable in Java Programming. Whether this statement is true or false?

Correct Answer

True

Solution Description

In java, its compulsory to initialise any local variable before using it because compiler don't assign any default/garbage value to variables.

## Predict the output

What is the output of the following code if the input is : 5 10 ?

Scanner s = new Scanner(System.in);

int a = s.nextInt();

int b = s.nextInt();

System.out.println(a+b);

Correct Answer

15

Solution Description

"s.nextInt()" scans and returns the next token as int. A token is part of entered line that is separated from other tokens by space, tab or newline. So when input line is : “5 10”, then s.nextInt() returns the first token i.e. “5” as int and s.nextInt() again returns the next token i.e. “10” as int.

## Predict The String output

What is the output of the following code if the input string is "Coding Ninjas"?

Scanner s = new Scanner(System.in);

String str;

str = s.next();

System.out.print(str);

Correct Answer

Coding

Solution Description

"s.next()" returns the next token as String. A token is part of entered line that is separated from other tokens by space, tab or newline. So when input line is - “Coding Ninjas” then s.next() returns the first token i.e. “Coding”

## Multiple inputs

What is the output of the following code if input is :

10 abc def

Scanner s = new Scanner(System.in);

int a = s.nextInt();

String str = s.next();

System.out.print(a);

System.out.println(str);

Correct Answer

10abc

Solution Description

"s.nextInt()" scans and returns the next token as int. A token is part of entered line that is separated from other tokens by space, tab or newline. So when input line is - “10 abc def” then s.nextInt() returns the first token as int i.e. “10” and s.next() returns the next token "abc”. While printing, in first statement a is printed and then str. There is no space or nextline between both print. Hence output is : 10abc.

## Integer and String

What is the output of the following code if input is : abc def 10

Scanner s = new Scanner(System.in);

String str = s.next();

int a = s.nextInt();

System.out.print(str + " " + a);

Correct Answer

InputMismatchException

Solution Description

"s.next()" scans and returns the next token as String. A token is part of entered line that is separated from other tokens by space, tab or newline. So when input line is - “abc def 10” then s.next() returns the first token as String i.e. “abc” and s.nextInt() tries to convert the next token i.e. “def” into an int, which gives InputMismatchException.

## Average Marks

Write a program to input a name(as a single character) and marks of three tests as m1, m2, and m3 of a student considering all the three marks have been given in integer format.

Now, you need to calculate the average of the given marks and print it along with the name as mentioned in the output format section.

All the test marks are in integers and hence calculate the average in integer as well. That is, you need to print the integer part of the average only and neglect the decimal part.

Input format :

Line 1 : Name(Single character)

Line 2 : Marks scored in the 3 tests separated by single space.

Output format :

First line of output prints the name of the student.

Second line of the output prints the average mark.

Constraints

Marks for each student lie in the range 0 to 100 (both inclusive)

Code:

import java.util.Scanner;

public class Solution {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

String name = sc.next();

int m1 = sc.nextInt();

int m2 = sc.nextInt();

int m3 = sc.nextInt();

int avg = (m1 + m2 + m3) / 3;

System.out.println(name);

System.out.println(avg);

}

}

# Lecture 3 : How is Data Stored?

## Short data type

Short data type has a minimum value of –

Correct Answer

-32,768

Solution Description

Short is of 16 bits. Here, 1 bit is used to store the sign and the remaining 15 bits for the value. So the smallest value a short data type can store is : - 2^15., where 2^15 is -32,768.

## Byte

Range of **byte** data type is ?

Correct Answer

-128 to 127

Solution Description

####Range of byte(8 bits) is : -2^7 to 2^7-1

## Check for Error

Will following statement give an error ?

float f = 1.4;

Correct Answer

Yes

Solution Description

####System treats all decimal numbers as double by default. So 1.4 is stored as double(which is of 8 bytes). When we will try put a double into a float(which is of 4 bytes), it will give you error.

## Figure out the Output

What will be the output of the following statement ?

System.out.println('a' + 1);

Answer

98

Solution Description

####When you add a character and an int, it will add the ASCII value of char ‘a’ i.e 97 and int 1. So ans will be 98.

## Figure out the Output

What will be the output ?

int i = 'c';

System.out.println(i);

Answer

99

Solution Description

When we put char ‘c’ into an int, its ASCII value will be put in the int i.e. 99.

## Automatic type conversion

Automatic type conversion in Java takes place when :

Correct Answer

Two types are compatible and size of destination is larger than the source type.

# Lecture 4 : Conditionals and Loops

## Error in code

What is the error in this code?

byte b = 50;

b = b \* 50;

Correct Answer

Solution Description

####In 2nd statement, b is multiplied with 50 where b is byte variable and 50 is an integer. So after the multiplication, result comes as an integer value which is 2500. Now we are trying to assign this integer to a byte variable, which will cause the error.

## Find the output

public class Solution{

public static void main(String [] args) {

double a = 6 / 4;

int b = 6 / 4;

double c = a + b;

System.out.println(c);

}

}

Correct Answer

Solution Description

####When 6 / 4 is performed, both the operands of / are integer. Hence answer will be an int i.e. 1. When we store it in a (which is double), value of a will be 1.0 and value of b will be 1. Thus a + b will be 2.0.

## Predict the output

public class Solution{

public static void main(String [] args) {

double a = 55.5;

int b = 55;

a = a % 10;

b = b % 10;

System.out.println(a + " " + b);

}

}

Correct Answer

Solution Description

####% operator gives remainder. So a % 10 will give us 5.5 and b % 10 will give us 5. Hence output is : 5.5 5

## Predict the output

public class Solution {

public static void main(String [] args) {

int var1 = 5;

int var2 = 6;

System.out.print(var1 > var2);

}

}

Correct Answer

Solution Description

####> is a relational operator. So it will give the result as true or false only. var1 is not greater than var2, hence result is false.

## Find output

Find the output of the following code:

public static void main(String args[])

{

int a=10,b=15;

if(a>b)

{

System.out.print("a ");

}

else

{

System.out.print("b ");

}

System.out.print("is greater");

}

Correct Answer

B is greater

Solution Description

"is greater" is written outside if-else so it would always print

## Find output

Find the output of the following code:

int a=50;

if(a>10)

{

System.out.print("Coding");

}

else(a>20)

{

System.out.print("Ninjas");

}

Correct Answer

Compile time error

Solution Description

else (a>20) is wrong syntax.

We cannot use condition after else.

## Find the output

public static void main(String args[])

{

int x = 5;

if (x < 6)

System.out.print("Hello ");

if(x == 5){

System.out.print("Hi ");

}

else{

System.out.print("Hey ");

}

}

Correct Answer

Hello Hi

Solution Description

As x is equal to 5, it enters in first if condition and prints "Hello". After that, second if condition will be checked and that is true again, so it will next print "Hi". Else part will be skipped.

## Figure out the output

public static void main(String[] args) {

int x = 15;

if(x <= 15){

System.out.print("Inside if ");

}else if(x == 15){

System.out.print("Inside else if ");

}

System.out.println(x);

}

Correct Answer

Inside if 15

Solution Description

####First if condition will be checked first, and it evaluates to true. Hence, the statement inside if will be executed, so it will print "Inside if ".

####After that, else if part will be skipped. Because in if-else statements, once a condition is satisfied remaining all the conditions are skipped without evaluation.

####After exit from if-else statement, the last print statement will be executed. Hence it will next print value of x i.e. 15.

####So the output is : "Inside if 15"

## What will be the output

public static void main(String args[])

{

int var1 = 5;

int var2 = 6;

if ((var2 = 1) == var1)

System.out.print(var2);

else

System.out.print(var2 + 1);

}

Correct Answer

2

Solution Description

####Inside if condition, we are actually assigning 1 to var2 and then comparing it with var1. So, after assignment, var2 becomes equal to 1 and then we are comparing it with var1 whose value is 5. Both are not equal, hence else part will be executed.

####So the output will be 2, as var2 was updated to value 1 in the if condition.

## Positive Negative

Let a and b are the two integers. Which option can be used to check out that one of the numbers is positive and the other is negative?

[Hint: product of one negative and one positive number will be always negative]

Correct Answer

a\*b<0

Solution Description

Given two integers a and b, the product of two integers is negative means either of a or b is negative.

i.e, If we multiply a\*b and the result is less than 0 , it means either a or b is negative.

## Find the output

Find the output of the following code:

public static void main (String[] args) {

int a=50;

int b=5;

if(a/0==b)

{

System.out.println("Hello");

}

else

{

System.out.println("Hi");

}

}

Correct Answer

Run time error

Solution Description

The syntax of the code is correct but on execution dividing a number by zero gives a run time error.

## Find the output

Find the output of the code :

main() {

int a=50,b=20;

if(a>b)

{

if(a>100)

print ("Ace");

if(b<100)

b=50;

}

else if(a==b)

{

print ("King");

}

else

{

print ("Queen");

}

}

Correct Answer

No Output

Solution Description

(a>b) is true,So we go inside the if.

(a>100) is false.

(b<100) is true, so it is executed and b is updated to 50.

But no print statement is executed hence no output.

## Find Character Case

Write a program that takes a character as input and prints either 1, 0 or -1 according to the following rules.

1, if the character is an uppercase alphabet (A - Z)

0, if the character is a lowercase alphabet (a - z)

-1, if the character is not an alphabet

Input format :

Single Character

Output format :

1 or 0 or -1

Constraints :

Input can be any character

Sample Input 1 :

v

Sample Output 1 :

0

Sample Input 2 :

V

Sample Output 2 :

1

Sample Input 3 :

#

Sample Output 3 :

-1

CODE:

import java.util.Scanner;

public class Solution {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

char ch = scanner.next().charAt(0);

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

System.out.println("1");

} else if (ch >= 'a' && ch <= 'z') {

System.out.println("0");

} else {

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

}

}

}

## Find the output

What will be the output of the following code:

public static void main (String[] args) {

int i=0;

while(i<10)

{

i=i+1;

System.out.print(i);

i=i+1;

}

}

Correct Answer

Solution Description

The variable i =0. First time the loop executes, it increases the variable by 1 , print is and then again increases it by 1, making it 2. Now, second time, since the condition is true, the variable i would be incaresed to 1 , ie. i=3 and would be printed and so on the loop will continue, till i=9. So the output would be all the odd numbers<10, which is : 13579

## Number of hello

The number of Hello printed on the screen for the following code will be:

public static void main (String[] args) {

int x=5;

int y=5;

while((x=5)==y)

{

System.out.println("Hello");

x++;

y++;

}

}

Correct Answer

One

Solution Description

The loop will run the first time as x=5 and y is also 5. Both are equal so the loop will get executed and after printing "Hello" x and y both will be incremented to 6. Now, if we check the while condition x=5 (x is being assigned the value 5) and y=6 (due to increment). So, x and y are not equal , hence condition is false and the loop will not get executed.

## Number of hello 2

The number of Hello printed on the screen for the following code will be:

public static void main (String[] args) {

int x=5;

int y=5;

while(x==y)

{

System.out.println("Hello");

x++;

y++;

}

}

Correct Answer

Solution Description

x and y are equal every time . Hence infinite loop.

## Same output

Which of the following codes gives same output:

1

int i=1;

while(i<5)

{

System.out.print(2\*i);

i=i+1;

}

2

int i=2;

while(i<10)

{

System.out.print(i);

i=i+2;

}

3

int i=2;

while(i<10)

{

System.out.print(i);

i\*=2;

}

4

int i=10;

while(i>0)

{

if(i%2==0)

{

System.out.print(10-i);

}

i--;

}

Correct Answer

1 and 2

Solution Description

Output of the codes are:

Code 1: 2468

Code 2: 2468

Code 3: 248

Code 4: 02468

## Fahrenheit to Celsius Table

Given three values - Start Fahrenheit Value (S), End Fahrenheit value (E) and Step Size (W), you need to convert all Fahrenheit values from Start to End at the gap of W, into their corresponding Celsius values and print the table.

Input Format :

3 integers - S, E and W respectively

Output Format :

Fahrenheit to Celsius conversion table. One line for every Fahrenheit and corresponding Celsius value in integer form. The Fahrenheit value and its corresponding Celsius value should be separate by single space.

Constraints :

0 <= S <= 90

S <= E <= 900

0 <= W <= 80

Sample Input 1:

0

100

20

Sample Output 1:

0 -17

20 -6

40 4

60 15

80 26

100 37

Sample Input 2:

20

119

13

Sample Output 2:

20 -6

33 0

46 7

59 15

72 22

85 29

98 36

111 43

Explanation For Input 2:

Start calculating the Celsius values for each Fahrenheit Value which starts from 20. So starting from 20, we need to compute its corresponding Celsius value which computes to -6. We print this information as <Fahrenheit Value> <a single space> <Celsius Value> on each line. Now add 13 to Fahrenheit Value at each step until you reach 119 in this case. You may or may not exactly land on the end value depending on the steps you are taking.

CODE:

import java.util.Scanner;

public class Solution {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

int S = scanner.nextInt();

int E = scanner.nextInt();

int W = scanner.nextInt();

int fahrenheit;

while(S<=E){

fahrenheit=((5\*(S-32))/9);

System.out.println(S+" "+fahrenheit);

S+=W;

}

}

}

## Primality Check

Can this code be used to check primality of a positive integer:

public static void main (String[] args) {

Scanner s=new Scanner (System.in);

int n=s.nextInt();

boolean isprime=true;

if(n%2==0)

isprime=false;

int i=3;

while(isprime&&i<n)

{

isprime=!(n%i==0);

i+=2;

}

if(isprime)

{

System.out.println("Prime");

}

else

{

System.out.println("Composite");

}

}

Correct Answer

No

Solution Description

The code fails for n=2. Otherwise the code runs fine for every other positive integer.

## Find the output

Find the output for the following code:

int i=10;

while((i=i-1)>0)

{

System.out.print(i);

if(i%5==0)

return;

}

Correct Answer

98765

Solution Description

The code executes for i=9,8,7,6,5 and prints 98765.

At i=5 the if statement is executed and the code terminates.

## Check the error

Will following code generate error ?

public class Main {

public static void main(String[] args) {

int a = 10;

if(a > 5) {

int b = 10;

}

System.out.println(b);

}

}

Correct Answer

Yes

Solution Description

Since int b is declared within the if block, it will not be accessible beyond it. We can see that the statement to print b is written outside the if block, hence it would generate error.

## Check the error

Will following code generate error ?

public class Main {

public static void main(String[] args) {

int a = 10;

if(a > 5) {

a = 100;

}

System.out.println(a);

}

}

Correct Answer

No

Solution Description

The variable a is accessible within the complete main (). So, the code will not generate any error.

## Check the error

Will following code generate error ?

public class Main {

public static void main(String[] args) {

int a = 10;

if(a > 5) {

int a = 100;

}

System.out.println(a);

}

}

Correct Answer

Yes

Solution Description

Java does not allow us to declare a same variable twice. We can see that int a was declared in the main() and then again it was declared inside the if block, which is not permitted. So, the code will throw the error.

## Check the error

Will following code generate error ?

public class Main {

public static void main(String[] args) {

for(int i = 0; i < 3; i++) {

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

}

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

}

}

Correct Answer

Yes

## What is the output

What is the output ?

public class Main {

public static void main(String[] args) {

int a = 10;

while(a > 5) {

int b = 1;

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

a--;

}

}

}

Correct Answer

1 1 1 1 1

Solution Description

The while loop will be executed 5 times, starting from when a=10 till it become a=6/. Each time the variable b since it is inside the while, would be initialized by 1, so the output will be 1 1 1 1 1

# Assignment

## Total Salary

Write a program to calculate the total salary of a person. The user has to enter the basic salary (an integer) and the grade (an uppercase character), and depending upon which the total salary is calculated as -

totalSalary = basic + hra + da + allow – pf

where :

hra = 20% of basic

da = 50% of basic

allow = 1700 if grade = ‘A’

allow = 1500 if grade = ‘B’

allow = 1300 if grade = ‘C' or any other character

pf = 11% of basic.

Round off the total salary and then print the integral part only.

Note for C++ users :

To round off the value , please include<cmath> library in the start of the program.

And round off the values in this way

int ans = round(yourFinalValue);

Input format :

Basic salary & Grade (separated by space)

Output Format :

Total Salary

Constraints :

0 <= Basic Salary <= 7,500,000

Sample Input 1 :

10000 A

Sample Output 1 :

17600

Sample Input 2 :

4567 B

Sample Output 2 :

8762

Explanation of Input 2:

We have been given the basic salary as Rs. 4567. We need to calculate the hra, da and pf.

Now when we calculate each of the, it turns out to be:

hra = 20% of Rs. 4567 = Rs. 913.4

da = 50% od Rs. 4567 = Rs. 2283.5

pf = 11% of Rs. 4567 = Rs. 502.37

Since, the grade is 'B', we take allowance as Rs. 1500.

On substituting these values to the formula of totalSalary, we get Rs. 8761.53 and now rounding it off will result in Rs. 8762 and hence the Answer.

CODE:

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

int basicSalary = input.nextInt();

char grade = input.next().charAt(0);

double hra = 0.2 \* basicSalary;

double da = 0.5 \* basicSalary;

double allow = 0;

if (grade == 'A') {

allow = 1700;

} else if (grade == 'B') {

allow = 1500;

} else {

allow = 1300;

}

double pf = 0.11 \* basicSalary;

double totalSalary = basicSalary + hra + da + allow - pf;

int roundedSalary = (int) Math.round(totalSalary);

System.out.println(roundedSalary);

}

}

## Multiplication Table

Write a program to print multiplication table of n

Input Format :

A single integer, n

Output Format :

First 10 multiples of n each printed in new line

Constraints :

0 <= n <= 10,000

Sample Input 1 :

2

Sample Output 1 :

2

4

6

8

10

12

14

16

18

20

Sample Input 2 :

5

Sample Output 2 :

5

10

15

20

25

30

35

40

45

50

CODE:

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

int n=input.nextInt();

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

System.out.println(n\*i);

}

}

}

## Sum of even & odd

Write a program to input an integer N and print the sum of all its even digits and sum of all its odd digits separately.

Digits mean numbers, not the places! That is, if the given integer is "13245", even digits are 2 & 4 and odd digits are 1, 3 & 5.

Input format :

Integer N

Output format :

Sum\_of\_Even\_Digits Sum\_of\_Odd\_Digits

(Print first even sum and then odd sum separated by space)

Constraints

0 <= N <= 10^8

Sample Input 1:

1234

Sample Output 1:

6 4

Sample Input 2:

552245

Sample Output 2:

8 15

CODE:

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

int n = input.nextInt();

int evenSum = 0;

int oddSum = 0;

while (n != 0) {

int digit = n % 10;

if (digit % 2 == 0) {

evenSum += digit;

} else {

oddSum += digit;

}

n /= 10;

}

System.out.println(evenSum + " " + oddSum);

}

}

## Factors

Write a program that takes a number as input and prints all its factors. If the number has only two factors (1 and the number itself), then the program should print -1.

Input Format :

A single integer, n

Output Format :

All the factors of n excluding 1 and the number itself

Constraints :

0 <= n <= 10,000

Sample Input 1 :

8

Sample Output 1 :

2 4

Sample Input 2 :

11

Sample Output 2 :

-1

Explanation of Sample Output 2 :

11 is a prime number having factors 1 and 11 so that output will be -1.

CODE:

import java.util.\*;

public class Solution {

public static void main(String[] args) {

Scanner s=new Scanner(System.in);

int n =s.nextInt();

for (int i = 2; i < n; ++i) {

if (n % i == 0) {

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

}

}

}

}

## Find power of a number

Write a program to find x to the power n (i.e. x^n). Take x and n from the user. You need to print the answer.

Note : For this question, you can assume that 0 raised to the power of 0 is 1

Input format :

Two integers x and n (separated by space)

Output Format :

x^n (i.e. x raise to the power n)

Constraints:

0 <= x <= 8

0 <= n <= 9

Sample Input 1 :

3 4

Sample Output 1 :

81

Sample Input 2 :

2 5

Sample Output 2 :

32

CODE:

import java.util.Scanner;

public class Solution {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

int x = input.nextInt();

int n = input.nextInt();

int result = 1;

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

result \*= x;

}

System.out.println(result);

}

}

# Lecture 5 : Patterns 1

## Code : Square Pattern

Print the following pattern for the given N number of rows.

Pattern for N = 4

4444

4444

4444

4444

Input format :

Integer N (Total no. of rows)

Output format :

Pattern in N lines

Constraints

0 <= N <= 50

Sample Input 1:

7

Sample Output 1:

7777777

7777777

7777777

7777777

7777777

7777777

7777777

Sample Input 1:

6

Sample Output 1:

666666

666666

666666

666666

666666

666666

CODE:

import java.util.\*;

public class Solution {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int N = sc.nextInt();

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

for (int j = 0; j < N; j++) {

System.out.print(N);

}

System.out.println("");

}

}

}

## Code : Triangular Star Pattern

Print the following pattern for the given N number of rows.

Pattern for N = 4

\*

\*\*

\*\*\*

\*\*\*\*

Note : There are no spaces between the stars (\*).

Input format :

Integer N (Total no. of rows)

Output format :

Pattern in N lines

Constraints

0 <= N <= 50

Sample Input 1:

5

Sample Output 1:

\*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*

Sample Input 2:

6

Sample Output 2:

\*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*

CODE:

import java.util.\*;

public class Solution {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int N = sc.nextInt();

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

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

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

}

System.out.println("");

}

}

}

## Code : Triangle Number Pattern

Print the following pattern for the given N number of rows.

Pattern for N = 4

1

22

333

4444

Input format :

Integer N (Total no. of rows)

Output format :

Pattern in N lines

Constraints

0 <= N <= 50

Sample Input 1:

5

Sample Output 1:

1

22

333

4444

55555

Sample Input 2:

6

Sample Output 2:

1

22

333

4444

55555

666666

CODE:

import java.util.\*;

public class Solution {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int N = sc.nextInt();

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

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

System.out.print(i);

}

System.out.println("");

}

}

}

## Code : Reverse Number Pattern

Print the following pattern for the given N number of rows.

Pattern for N = 4

1

21

321

4321

Input format :

Integer N (Total no. of rows)

Output format :

Pattern in N lines

Constraints

0 <= N <= 50

Sample Input 1:

5

Sample Output 1:

1

21

321

4321

54321

Sample Input 2:

6

Sample Output 2:

1

21

321

4321

54321

654321

CODE:

import java.util.\*;

public class Solution {

public static void main(String[] args) {

Scanner sc= new Scanner(System.in);

int N = sc.nextInt();

int m;

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

{

m=i;

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

{

System.out.print(m);

m--;

}

System.out.println("");

}

}

}

## Code : Alpha Pattern

Print the following pattern for the given N number of rows.

Pattern for N = 3

A

BB

CCC

Input format :

Integer N (Total no. of rows)

Output format :

Pattern in N lines

Constraints

0 <= N <= 26

Sample Input 1:

7

Sample Output 1:

A

BB

CCC

DDDD

EEEEE

FFFFFF

GGGGGGG

Sample Input 2:

6

Sample Output 2:

A

BB

CCC

DDDD

EEEEE

FFFFFF

CODE:

import java.util.\*;

public class Solution {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int N = sc.nextInt();

char c = 65;

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

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

System.out.print(c);

}

c++;

System.out.println("");

}

}

}

## Code : Character Pattern

Print the following pattern for the given N number of rows.

Pattern for N = 4

A

BC

CDE

DEFG

Input format :

Integer N (Total no. of rows)

Output format :

Pattern in N lines

Constraints

0 <= N <= 13

Sample Input 1:

5

Sample Output 1:

A

BC

CDE

DEFG

EFGHI

Sample Input 2:

6

Sample Output 2:

A

BC

CDE

DEFG

EFGHI

FGHIJK

CODE:

import java.util.\*;

public class Solution {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int N = sc.nextInt();

int m;

char c;

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

m = 64 + i;

c = (char) m;

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

System.out.print(c);

c++;

}

System.out.println("");

}

}

}

## Code : Interesting Alphabets

Print the following pattern for the given number of rows.

Pattern for N = 5

E

DE

CDE

BCDE

ABCDE

Input format :

N (Total no. of rows)

Output format :

Pattern in N lines

Constraints

0 <= N <= 26

Sample Input 1:

8

Sample Output 1:

H

GH

FGH

EFGH

DEFGH

CDEFGH

BCDEFGH

ABCDEFGH

Sample Input 2:

7

Sample Output 2:

G

FG

EFG

DEFG

CDEFG

BCDEFG

ABCDEFG

CODE:

import java.util.\*;

public class Solution {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int N = sc.nextInt();

int m;

char c = (char) (65 + N);

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

c = (char) (c - i);

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

System.out.print(c);

c++;

}

System.out.println("");

}

}

}

# Lecture 6 : Patterns 2

## Code : Mirror Image Number Pattern

Print the following pattern for the given N number of rows.

Pattern for N = 4

The dots represent spaces.

Input format :

Integer N (Total no. of rows)

Output format :

Pattern in N lines

Constraints

0 <= N <= 50

Sample Input 1:

3

Sample Output 1:

1

12

123

Sample Input 2:

4

Sample Output 2:

1

12

123

1234

CODE:

import java.util.\*;

public class Solution {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

int count = 1;

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

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

System.out.print(" ");

for (int j = 1; j <= count; j++) {

System.out.print(j);

}

System.out.println();

count++;

}

}

}

## Code : Inverted Number Pattern

Print the following pattern for the given N number of rows.

Pattern for N = 4

4444

333

22

1

Input format :

Integer N (Total no. of rows)

Output format :

Pattern in N lines

Constraints :

0 <= N <= 50

Sample Input 1:

5

Sample Output 1:

55555

4444

333

22

1

Sample Input 2:

6

Sample Output 2:

666666

55555

4444

333

22

1

CODE:

import java.util.\*;

public class Solution {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

for (int i = n; i >= 1; i--) {

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

System.out.print(i);

}

System.out.println();

}

}

}

## Code : Star Pattern

Print the following pattern

Pattern for N = 4

The dots represent spaces.

Input Format :

N (Total no. of rows)

Output Format :

Pattern in N lines

Constraints :

0 <= N <= 50

Sample Input 1 :

3

Sample Output 1 :

\*

\*\*\*

\*\*\*\*\*

Sample Input 2 :

4

Sample Output 2 :

\*

\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*\*

CODE:

import java.util.Scanner;

public class Solution {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

int space = n - 1;

int i = 1;

while (i <= n) {

int j = 1;

space = n - i;

while (space > 0) {

System.out.print(" ");

space--;

}

while (j <= ((2 \* i) - 1)) {

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

j += 1;

}

System.out.println("");

i += 1;

}

}

## Code : Triangle of Numbers

Print the following pattern for the given number of rows.

Pattern for N = 4

The dots represent spaces.

Input format :

Integer N (Total no. of rows)

Output format :

Pattern in N lines

Constraints :

0 <= N <= 50

Sample Input 1:

5

Sample Output 1:

1

232

34543

4567654

567898765

Sample Input 2:

4

Sample Output 2:

1

232

34543

4567654

CODE:

import java.util.Scanner;

public class Solution {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

int i, j, num = 1, gap;

gap = n - 1;

for (j = 1; j <= n; j++) {

num = j;

for (i = 1; i <= gap; i++)

System.out.print(" ");

gap--;

for (i = 1; i <= j; i++) {

System.out.print(num);

num++;

}

num--;

num--;

for (i = 1; i < j; i++) {

System.out.print(num);

num--;

}

System.out.println();

}

}

}

## Code : Diamond of stars

Print the following pattern for the given number of rows.

Note: N is always odd.

Pattern for N = 5

The dots represent spaces.

Input format :

N (Total no. of rows and can only be odd)

Output format :

Pattern in N lines

Constraints :

1 <= N <= 49

Sample Input 1:

5

Sample Output 1:

\*

\*\*\*

\*\*\*\*\*

\*\*\*

\*

Sample Input 2:

3

Sample Output 2:

\*

\*\*\*

\*

CODE:

import java.util.Scanner;

public class Solution {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int row = sc.nextInt();

int i, j, space, decider;

decider = row / 2;

for (i = 0; i < row; i++) {

if (i <= decider) {

for (space = i; space < decider; space++)

System.out.print(" ");

for (j = 0; j < (i \* 2) + 1; j++)

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

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

} else {

for (space = i; space > decider; space--)

System.out.print(" ");

for (j = (row \* 2) - 1; j > (i \* 2); j--)

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

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

}

}

}

}

# Assignment

## Half Diamond Pattern

Write a program to print N number of rows for Half Diamond pattern using stars and numbers

Note : There are no spaces between the characters in a single line.

Input Format :

A single integer: N

Output Format :

Required Pattern

Constraints :

0 <= N <= 50

Sample Input 1 :

3

Sample Output 1 :

\*

\*1\*

\*121\*

\*12321\*

\*121\*

\*1\*

\*

Sample Input 2 :

5

Sample Output 2 :

\*

\*1\*

\*121\*

\*12321\*

\*1234321\*

\*123454321\*

\*1234321\*

\*12321\*

\*121\*

\*1\*

\*

CODE:

import java.util.Scanner;

public class Solution {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

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

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

{

int j = 1;

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

while (j <= i)

{

System.out.print(j);

j++;

}

j--;

while (--j >= 1)

{

System.out.print(j );

}

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

}

for (int i = n - 1; i >= 1; i--)

{

int j = 1;

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

while (j <= i) {

System.out.print(j);

j++;

}

j--;

while (--j >= 1)

{

System.out.print(j );

}

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

}

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

}

}

## Parallelogram Pattern

Write a program to print parallelogram pattern for the given N number of rows.

For N = 4

The dots represent spaces.

Input Format :

A single integer : N

Output Format :

Required Pattern

Constraints :

0 <= N <= 50

Sample Input 1 :

3

Sample Output 1 :

\*\*\*

\*\*\*

\*\*\*

Sample Input 2 :

5

Sample Output 2 :

\*\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*

CODE:

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int row = sc.nextInt();

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

{

for (int j = 2; j <= i; j++)

{

System.out.print(" ");

}

for (int j = 1; j <= row; j++)

{

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

}

System.out.println();

}

}

}

## Sum Pattern

Write a program to print triangle of user defined integers sum.

Input Format :

A single integer, N

Output Format :

Required Pattern

Constraints :

0 <= N <= 50

Sample Input 1 :

3

Sample Output 1 :

1=1

1+2=3

1+2+3=6

Sample Input 2 :

5

Sample Output 2 :

1=1

1+2=3

1+2+3=6

1+2+3+4=10

1+2+3+4+5=15

CODE:

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

int i, j, num;

for(i=1; i<=n; i++)

{

num=1;

int sum=0;

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

{

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

sum=sum+num;

if (num<i)

{

System.out.print("+");

}

num++;

}

System.out.print("="+sum);

System.out.println();

}

}

}

## Odd Square

Write a program to print the pattern for the given N number of rows.

For N = 4

1357

3571

5713

7135

Input Format :

A single integer: N

Output Format :

Required Pattern

Constraints :

0 <= N <= 50

Sample Input 1 :

3

Sample Output 1 :

135

351

513

Sample Input 2 :

5

Sample Output 2 :

13579

35791

57913

79135

91357

CODE:

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

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

{

for (int j = i - 1; j < n; j++ )

{

System.out.print(j \* 2 + 1 + "");

}

for(int k = 0; k < i - 1; k++)

{

System.out.print(k \* 2 + 1 + "");

}

System.out.println();

}

}

}