

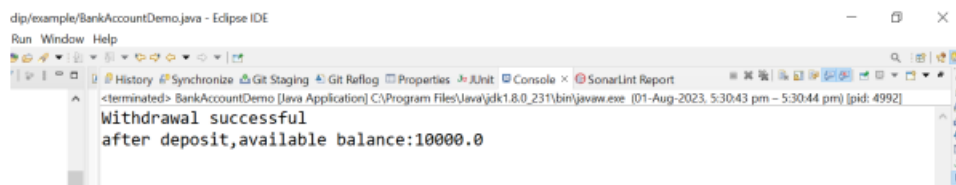
Lab 1.

Q 1

Create a Bank class and declare an instance variable named amount of type double. Create parameterized constructor to initialize variable "amount" with value 10000. Create two methods withdraw(double withdrawalAmount) and deposit(double depositAmount). Calculate withdrawal based on some condition (using ternary operator) like If amount is sufficient then "withdraw successful" message will be printed on the console and amount should be updated after withdraw. Later on, deposit 5000 in the account balance. At the end display total balance on the console. String message = (withdrawalAmount <= amount) ? "Withdrawal successful" : "Insufficient balance"; [Hint: Use constructor, ternary operator] Sample input: Amount=10000

Withdrawal amount=5000 Deposit amount=5000 Expected output:

Expected output:



```
dip/example/BankAccountDemo.java - Eclipse IDE
Run Window Help
History Synchronize Git Staging Git Reflog Properties JUnit Console SonarLint Report
<terminated> BankAccountDemo [Java Application] C:\Program Files\Java\jdk1.8.0_231\bin\java.exe (01-Aug-2023, 5:30:43 pm - 5:30:44 pm) [pid: 4992]
Withdrawal successful
after deposit, available balance: 10000.0
```

Q 2

2. Write a program to input two numbers and find the maximum between two numbers using the conditional/ternary operator.

Sample Input:

num1 = 10
num2 = 30

Expected Output:



```
snudip/example/Assignment.java - Eclipse IDE
Run Window Help
History Synchronize Git Staging Git Reflog Properties JUnit Console SonarLint Report
<terminated> Assignment [Java Application] C:\Program Files\Java\jdk1.8.0_231\bin\java.exe (01-Aug-2023, 5:45:51 pm - 5:45:53 pm) [pid: 16532]
The maximum between 10 and 30 is: 30
```

3. Write a program to declare two variables num and n and take an input during compilation time to check whether the nth bit of the given number is **set (1)** or **not (0)**.

Logic to get nth bit of a number

Step by step descriptive logic to get the nth bit of a number.

1. Take an input of any number and Store it in some variable, say **num**.
2. Take an Input the bit position and Store it in some variable, say **n**.
3. To get the nth bit of num right shift num, n times. Then perform bitwise AND with 1 i.e. bitStatus = (num >> n) & 1.

Sample Input:

Input number: num=12

Input nth bit number: n=2

Expected output:

