**Section A**

1. Automatic type conversion is impossible in which of the possible cases?

* byte to int
* int to long
* ***long to int***
* short to int

1. Find the output of the following code

|  |
| --- |
| public class Solution{        public static void main(String[] args){                int[]  x = {120, 200, 016};                for(int i = 0; i < x.length; i++){                         System.out.print(x[i] + " ");                }                           } } |

* 120 200 016
* 120 200 16
* ***120 200 14***
* Compileerror

1. Find the output of the following code

|  |
| --- |
| public class Solution{        public static void main(String[] args){                      short x = 10;                      x =  x \* 5;                      System.out.print(x);        } } |

* 50
* 10
* ***Compile error***
* Exception

1. When an array is passed to a method, what does the method receive?

* ***The reference of the array***
* A copy of the array
* Length of the array
* Copy of first element

1. Find the output of the following code

|  |
| --- |
| int Integer = 24; char String  = 'I'; System.out.print(Integer); System.out.print(String); |

* Compile error
* Throws Exception
* I
* ***24 I***

1. Which among the following is a valid expression to declare and initialise an array,

* int[] A = {}
* ***int[] A = {1,2,3}***
* int[] A = (1,2,3)
* int[][] A = {1,2,3}

1. Identify the keyword among the following that makes a variable belong to a class,rather than being defined for each instance of the class.

* final
* ***static***
* volatile
* abstract

1. Which of the following statements are true about finalize() method?

* ***It can be called zero or one time***
* It can be called zero or more times
* It can be called exactly once
* It can be called multiple times based on the call

1. Find the output of the following code

|  |
| --- |
| public class Solution{          public static void main(String args[]){                  int i;                  for(i = 1; i < 6; i++){                       if(i > 3) continue;                  }                  System.out.println(i);           } } |

* 3
* 4
* 5
* ***6***

1. Identify which of the following can directly access and change the value of the variable ***result*** ?

|  |
| --- |
| package org.prograd; public class Solution{        private int res = 100; } |

* Any class
* ***Only the Solution class***
* Any class that extends Solution class
* None of the above

1. Find A[1] in the following code

|  |
| --- |
| int[] A = {0,2,4,1,3}; for(int i = 0; i < a.length; i++){     a[i] = a[(a[i] + 3) % a.length]; }  System.out.println(A[1]); |

* 0
* ***1***
* 2
* 3

1. Arrays in Java are,

* ***Objects***
* Object references
* Primitives
* None of the above

1. Find the output of the following code

|  |
| --- |
| public class Solution{        public static void main(String[] args){                      byte x = 127;                      x++;                      x++;                      System.out.print(x);        } } |

* 127
* 129
* ***-127***
* 2

1. Which of the following is used to find and fix bugs in the program?

* JDK
* JRE
* JVM
* ***JDB***

1. How many times will “ProGrad” be printed?

|  |
| --- |
| int count = 0; do{   System.out.println("ProGrad");   count++; } while(count < 10); |

* 8
* 9
* ***10***
* 11

**Section B**

1. Raju faces some difficulties in converting numbers to Roman numerals and vice versa. Make his life easier by creating a converter that helps him convert numbers in both directions. (PS: Use TDD approach and create an optimised converter)

1. There were two good friends, ***Hero*** and ***Villain***. Both had the same interest and same love towards various types of weapons. Both wanted to be the most skillful person ever in handling any weapon. Both used to practise together and challenge each other. Most of the time the battle used to be on a fun side, but this time it turned out to be a serious one. Villain was less skillful than ***Hero***, but still ***Villain*** used to win most of the battles, by getting some unfair advantage. ***Hero*** being a nice person, never used to complain about this behaviour of ***Villain***. But today, he couldn't bear anymore, so ***Hero*** challenged ***Villain*** for a real fight, with any weapon of their choice. But both having same interest of weapons, unknowingly chose the same weapons -

|  |
| --- |
| gunOne - damage done = 30 gunTwo - damage done = 50 [sword, spears] |

The interesting part is that, the only damage that can happen - can only happen through either *gunOne* or *gunTwo*, because both are equally good at handling swords or spears.

***Villain*** being ***Villain*** -- seeking for some unfair advantage, he thought of wearing an armour inside his clothes. This armour reduces the damage done by each gun by 10 -->

|  |
| --- |
| gunOne - damage with armour = 20 gunTwo - damage with armour = 40 |

Your task is to create a system, where the above scenario can be shown.

Add on : Seeing ***Villain*** taking this unfair advantage of the shield/armour; our ***Hero*** is also planning to make one. But the issue is that - the ***Hero*** is not aware of how to make one. So, for now it is an abstract case for him, which for sure he wants to make into existence.

1. Candy crush, being the world famous game, its investors wanted to include the feature of Global leaderboard. You are expected to suggest to them the optimal algorithm and the best logic to be implemented. The constraints are,

* There are a few millions of users and their highscores already stored in the servers,
* The top 10 are expected to be displayed on the leaderboard.
* Everytime, when a new player finishes a game, the new score has to be compared with the existing highscores, and placed properly in the leaderboard.

1. Lifts have become unavoidable in today’s modernised world. Let’s try to write a (TDD) software based on the working of a lift. The expectations of a lift system are,

* The lift should have an attribute floor, indicating the *currentFloor*
* The lift responds to the user’s calls, that includes the *sourceFloor*. It is also expected have the attribute *direction* (to be calculated automatically based on the *currentFloor* and *sourceFloor* attributes)
* The lift has to take the users to the *destinationFloor* (with the attribute *direction* - same as the previous attribute)

You are also expected to implement the following,

* The *currentFloor* attribute,
* The *Opening & Closing* actions of the lift door,
* The signature *“Ding” sounds* every time, after reaching the destination floors.

Add ons:

* There can be more than one lifts
* A lift doesn’t respond immediately; Consider including a waiting time/multiple parallel user calls.
* Queuing the user calls, executed in a particular sequence.

1. Banks are fed-up using ALGOL and COBOL for their mainframes. You are recruited to create a prototype of the whole banking process via your favourite programming language, Java! Let's show your expertise. The expected modules are,

* To create a new account holder (CRUD),
* To add new debit card/cheque book for the existing account holders (CRUD),
* To perform the daily actions with the bank account like,
  + Checking balance,
  + Depositing money,
  + Withdrawing money.

Add ons:

* Enabling account-to-account transfer to existing account holders (CRUD ,
* Account to account Money transfer can also be included in the existing system.