OOP

**Q.1 How would you describe the abstraction in OOP?**

Abstraction is a process of hiding the implementation details and showing only functionality to the user.

Ex. Let’s assume you will create a block of code which takes something as input and gives out some other data as output. Abstraction means that other users or developers who deal with input and output don’t need to know what happens in your block. This also allows you to change or rewrite your code in a way where the input and output will remain the same.

**Q.2 How it is possible to implement Encapsulation in a class?**

make methods and variables inside of a class private

**Q.3 What are the benefits of Inheritance between classes?**

You can make so called subclasses which inherit methods and variables from so called upper class. Need to use keyword extends.

Main benefit is code reuse, you can extend the upper class to multiple subclasses without rewriting the code all the time

**Q.4 How would you describe the polymorphism?**

Something can morph into another object. One object can become another object.

Ex. We have upperclass Person – has subclasses student and teacher. We can treat student as a student but also as a person. So we can make:

Person person = new Student() or Student student = new Student().

**Q.5 What it means and where can be used the keyword static?**

It means that a variable or method can be accessed without requiring an instantiation of the class to which it belongs. You can call the method without creating an object to which it belongs.

Use when a common operation is required to perform throughout the class

**Q.6 Where can I access a protected method?**

You have to be in the same package. Or you have to be a subclass of the class where the protected method is described.

**Q.7 What it means for a method to be public or private?**

can't access private methods and variables outside of the class, need to use getters and setters to manipulate.

public methods are accessible from everywhere

**Q.8 What is abstract class?**

Abstract classes exist to be extended. Used on a class which have potentially many possible subclass objects.

Ex. We have class GameObject, which can have multiple potential subclasses – player, item, enemy and so on. So w emake GameObject class to abstract.

**Q.9 What is interface?**

An interface is 100% abstract class and has only abstract methods. A class can implement any number of interfaces.

Example we have class Mammal and we have class Pet. When we create a class Dog it cannot extend multiple upperclasses, so we need to imply interfaces instead.

**Q.10 Can you describe what is method override?**

For example if we have some method described in upper classes we can use override to change these methods to use in subclasses . Usually the subclass redefine the same method as the upperclass.

Threads:

1) What is a thread?

2) How can I run a new thread?

3) What does the synchronized modifier do when applied to a method?

4) What happens if I place synchronized on a static method?

5) What are race conditions between threads?

Collections

1. Describe Collections classes Api (List, Set, Map, Queue, Stack)

List is ordered collection and can contain duplicate elements.

Set is a collection that cannoc contain duplicate elements

Map is an object that maps keys to values

Queue

Stack

2) What are main implementations of List? (What are their difference)

3) Describe some implementations of Set interface

4) Where is hashcode method used in collections?

5) Describe how HashTable works (put and get)

6) What are buckets in HashMap?

7) What is the load factor in a HashMap?

8) How a TreeSet is able to order objects from a Person class?

9) How a Set<Person> is able to detect duplicates when I add a new Person in the Set?

10) What it means O(1) O(n) O(n2)?

JDBC

1) Can you describe main java objects involved into a task where I need to query DB tables?

2) Why Connections are expensive to cre