Spring 2019

NOTES

3(a)

Three principles of object oriented programming (oop) are:

i) Encapsulation
ii) Inheritance
iii) polymorphism

Encapsulation: Encapsulation in Java is a mechanism of wrapping the data (variables) and code acting on the data (methods) together as a single unit. ~~In~~

In encapsulation, the variables of a class will be hidden from other classes, and can be accessed only through the methods of their current class.

It makes the application simple and easy to debug. We can change

DATE

and make ~~the~~ application ~~simple and easy~~ edits to our codebase without disrupting the normal functioning of our program. The hygiene of a code is maintained easily.

It is also protects an object from unwanted access by clients. Reduces human errors and simplifies the maintance of the application. Also makes the application easier to understand.

Inheritance:

Inheritance in Java is a concept that acquires the properties from one class to other classes; for example, the relationship between father and son.

In Java, a class can inherit attributes and methods from another class. The class that inherits the properties is known as the sub class or the child class. The class from which the properties are inherited in known as the superclass or the parent class.

Syntax: Using extends keyword.

~~class Subclass-name~~

DATE

```
class Subclass-name extends Superclass-name
{
  //methods and fields
}
```

The extended keyword extends a class and is an indicator that a class is being inherited by another class. When we say, class B extends a class A, it means that class B is inheritting the properties (methods, attributes) from class A. Here, class A is the superclass or parent class and class ~~B~~ B is the subclass or child class.

It helps to reuse the code. The child class may use the code defined in the parent class without

re-writing it. As the main code need not be written again, so it saves time and effort. It provides a clear model structure which is easy to understand.

The main purpose of inheritance in object oriented programming (OOP) is to give the user ability to change the behaviour of the libraries, without actually changing already working and debugged code.

DATE

Polymorphism:

Polymorphism is one of the object ~~oreen~~ oriented programming feature that allows us to perform a single action in different ways. It is the capability of a method to do different things based on the object that it is acting upon. In other words, it allows us to define one interface and have multiple implementations.

For example, lets say we have a class Animal that has a method sound(). Since this is a generic class so we can not give it a implementation like : Roar, Meow etc. We had to give a generic message.

```
public class Animal {
    ...
    public void sound() {
        System.out.println("Animal is making
                            sound");
    }
}
```

Now lets say, we have two sub classes of Animal class: Horse and Cat that extends ~~(see~~ Animal class. We can provide the implementation to the same method like this:

```
public class Horse extends Animal {
    ...
    @Override
    public void sound() {
        System.out.println("Neigh");
    }
}
```

DATE

and

```
public class Cat extends Animal {
...
    @Override
public void sound() {
System.out.println("Meow");
  }
}
```

As we can see that, although we had the common action for all the subclasses sound() but there were different ways to do the same action.

There are two types of polymorphism.

1) Static polymorphism or. compile-time polymorphism.

Dynamic Polymorphism

NOTES

Static polymorphism:

Methods use the same name but the parameter varies, represents the static polymorphism.

Method overloading is an example of static polymorphism. In overloading, the method function has a same name but different signatures. It is also known as compile time polymorphism because the decision of which method is to be called is made at compile time.

Three criteria are provided in which the parameter sets have to differ:

i) The parameters number should vary.

ii) The parameter types should be different.

DATE

- Different order of parameters. For example, if a method accepts a string and a long, while the other method accepts a long and a string. However, this type of order makes it difficult for the API to understand.

Example:

```
class SimpleCalculator {
    int add(int a, int b)
    {
        return a+b;
    }
int add(int a, int b, int c)
    {
        return a+b+c;
    }
}
```

```
public class Demo {
public static void main (String args[])
{
SimpleCalculator obj = new SimpleCalc();
System.out.println(obj.add(25,25,30));
}
}
```

Output : 50
80

Dynamic polymorphism :

Dynamic polymorphism is a process or mechanism in which a call to an overridden method is to resolve at runtime rather than compile time. It is also known as runtime polymorphism or dynamic method dispatch. We can achieve dynamic polymorphism by using the method overriding.

DATE

Example:

There are two classes, named Bike and Car. The car class extends the class Bike and overriding its method run(). The run() method is called by the reference variable of the parent class. Since the subclass method is overriding the parent class method. The sub class method is called at run time.

```
class Bike{
void run(){
System.out.println("running");
  }
}
```

~~class Splendor extends~~

```
class Car extends Bike{
void run(){
System.out.println("walking safely
                    at with 30km");
}
public static void main(String args[]){
}
Bike b = new Car(); //upcasting
b.run();
}
}
```

Output: Waking safely with 60 km.

Programmers code can be reused via Polymorphism. supports a single variable for multiple data types. Reduces coupling between different functionalities.

DATE

```
try {
student.CheckNumber(S1.getNumber());
}catch(NumberExeception e){
  System.out.println(e);
  }
 }
}
```

5(c)

In Java the main method is declared as public because, it has to be call by the JVM (Java virtual Machine). So, if main() is not public in Java, the JVM won't call it.

The main method is declared as static as it is directly call by the JVM without creating an

object of the class in which ~~the~~ it is declared. When java runtime starts, there is no object of class present. Thats why main method has to be static, so JVM can load the class into memory and call the main method.

6(c)

The finally block in Java is used to put important codes such as clean up code e.g. closing the file or closing the connection. The finally block executes whether exception rise or not and whether exception handled or not. A finally contains all the crucial statements regardless of the exception occurs or not.

DATE

Example:

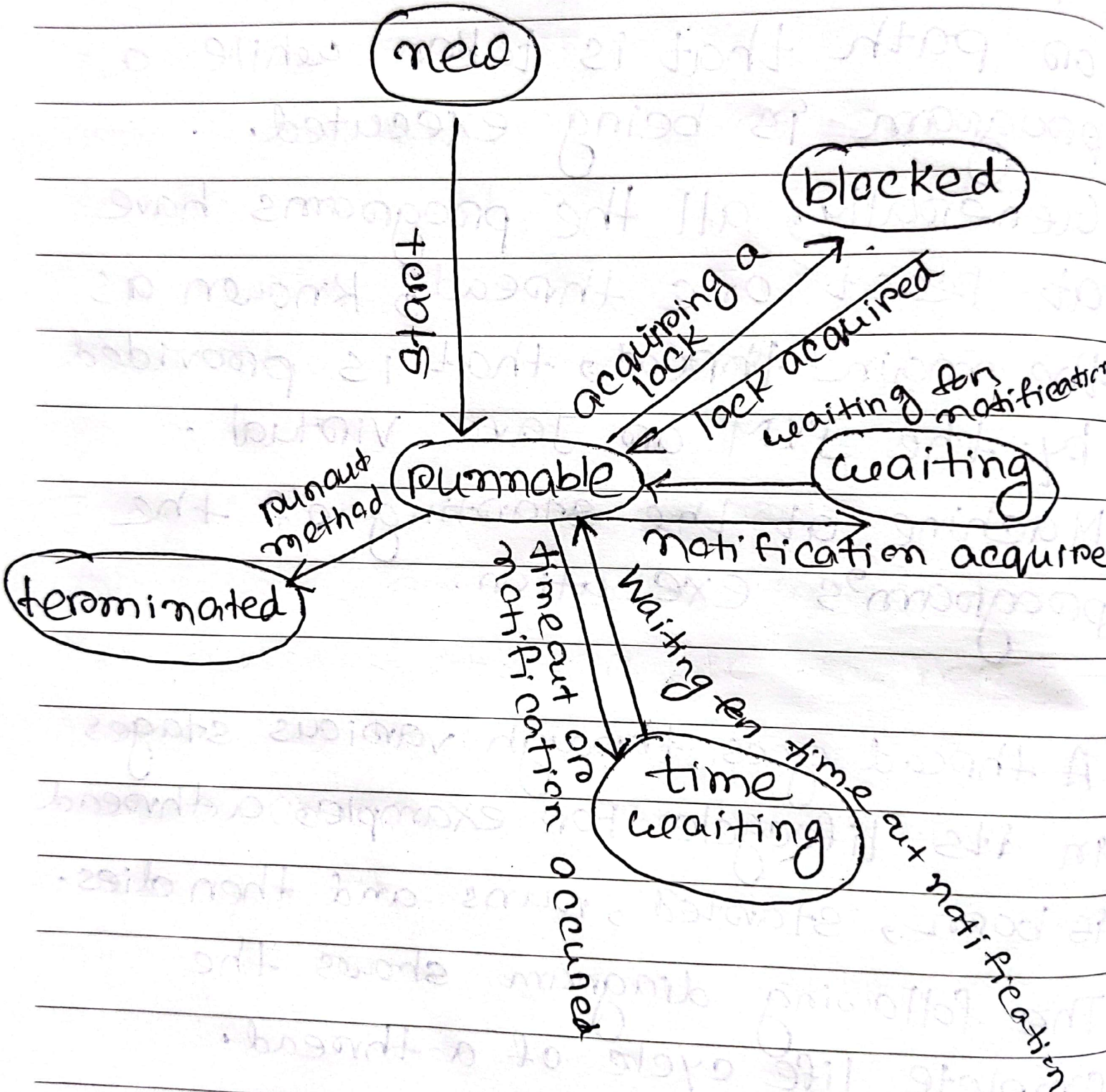
```
package code;
public class Test{
public static void main(String[] args){
 try{
System.out.println("testing");
 }finally
  {
   System.out.println("Last");
  }
 }
}
```

7(a)

A thread in Java is the direction or path that is taken while a program is being executed. Generally, all the programs have at least one thread, known as the main thread, that is provided by the JVM or Java Virtual Machine at the starting of the program's execution.

A thread goes through various stages in its lifecycle. For example, a thread is born, started, runs and then dies. The following diagram shows the complete life cycle of a thread.

DATE


new
start
blocked
acquiring a lock
lock acquired
waiting for notification
runnable
waiting
run out method
notification acquire
terminated
timeout or notification occures
waiting en time
time waiting
ex notification

A thread lies only in one of the shown states at any instant:

1. New
2. Runnable
3. Blocked
4. Waiting
5. Timed waiting
6. Terminated

1. New thread: When a new thread is created, it is in the new state. The thread has not yet started to run when the thread is in this state. When a thread lies in the new state, its code is yet to be run and hasn't started to execute.

DATE

2. Runnable state: A thread that is ready to run is moved to a runnable state. In this state, a thread might actually be running or it may be ready to run at any instant of time. It is the responsibility of the thread scheduler to give the thread, time to run. Each and every threads runs for a short while and then pauses and relinquishes the CPU to another thread so that other threads can get a chance to run. When this happens, all such threads that are ready to run, waiting for the CPU and the currently running thread lie in a runnable state.

3. Blocked / waiting state: When a thread is temporarily inactive, then it's in one of the following states:

Blocked
Waiting

4. Timed Waiting: A thread lies in a timed waiting state when it calls a method with a time-out parameter. A thread lies in this state until the timeout is completed or until a notification is received. For example, when a thread calls sleep or a conditional wait, it is moved to a timed waiting state.

DATE

5. Terminated State: A thread terminates because of either of the following reasons:

i) Because it exits normally. This happens when the code of the thread has been entirely executed by the program.

ii) Because there occurred some unusual erroneous event, like segmentation fault or an unhandled exception.

NOTES

7(b) A thread in java is the direction or path that is taken while a program is being executed.

Join method in java allows one thread to wait until another thread completes its execution. In simpler words, it means it waits for the other thread to die.

A join() is a final method of thread class and it can be used to join the start of a thread's execution to the end of another thread's execution so that a thread will not start running until another thread has ended.

DATE

Fall 2018

4c) What are the uses of this keyword?

Answer: The this keyword refers to the current object in a method or constructor. The most common use of the this keyword is to eliminate the confusion between class attributes and parameters with the same name because a class attribute is shadowed by a method or constructor parameter.

There are six uses of this keyword.

i) this can be used to get the current object.

ii) this can be used to invoke current object's method.

iii) this() can be used to invoke current class constructor.

iv) this can be passed as a parameter to a constructor.

v) this can be passed as a parameter to a method call.

vi) this can be used to return the current object from the method.

NOTES

Q(a): Which are the two ways that a thread can be created in Java?

Answer: A thread in Java is the direction or path that is taken while a program ~~is~~ is being executed. It doesn't block the user because threads are independent and a user can perform multiple operations at the same time.

There are two ways to create a thread:

1. By extending thread class.
2. By implementing Runnable interface.

When we extend thread class, each of our thread creates unique object and associate with it. When we implements Runnable, it shares

DATE

the same object to multiple threads.

Java only supports single inheritance so we can only extend one class.

Instantiating an interface gives a cleaner separation between our code and the implementation of threads. Implementing Runnable makes our class more flexible. If we extend thread then the action we are doing is always going to be in a thread.

So, implementing the runnable interface is preferable than extending thread class.

Spring 2018

NOTES

1)b: What is a class and what is an object? Provid a realistic example of a class. Your class has to have some instance variables and methods. Write down java code for your code.

Answer:

In the context of Java, a class is a templete used to create objects and to define object data types and methods.

In the context of Java, an object is a member or instance of a class. Each object has an identity, a behavior and a state.

DATE

c) How a java program is platform independent?

Answer:

Platform independent means implementing a code irrespective of the operating system on which it is being implemented.

Platform independent means it first converts the program to the intermediate state Byte code, which is platform independent and then compiles the program to another language source code.

Java is platform-independent because it uses a virtual machine. The Java programming language and all APIs are compiled into bytecodes. Bytecodes are effectively platform-independent. The virtual machine

takes care of the differences between the bytecodes for the different platforms. The run-time requirements for java are therefore very small. The java virtual machine takes care of all hardware-related issues, so that no code has to be complied for different hardware.

#What is automatic garbage collection in Java?

Answer:

Automatic garbage collection is a garbage collection technique in which the garbage collector runs automatically, without the programmer having to write code for it. In Java, garbage collection is done by putting the Java Virtual Machine

DATE

into a special mode. The garbage collector will run when it sees an opportunity to do so. Some garbage collectors use a stop-the-world approach, and the Java Virtual Machine must stop executing the program to do garbage collection. ~~Garbage collection is not an easy concept.~~

3a) Why a constructor is important? Should a programmers use no-argument constructor or a parameterized constructor or both?

Answer:

A constructor in Java is a special method that is used to initialize objects. In class-based object-

NOTES

oriented programming, a constructor is a special type of subroutine called to create an object. It prepares the new object for use, often accepting arguments that the constructor uses to set required member variables.

The sole purpose of the constructor is to initialize the data fields of objects in the class. Java constructor can perform any action but specially designed to perform initializing actions, such as initializing the instance variables. A constructor within a class allows constructing the object of the class at runtime.

DATE

A constructor that has no parameter is known as the default constructor. If we don't define a constructor in a class, then the compiler creates a default constructor with no arguments for the class. The compiler automatically provides a public no-argument constructor for any class without constructors.

A constructor is called Parameterized constructor when it accepts a specific number of parameters. To initialize data members of a class with distinct values. ~~In the above~~ It can exist even without the existence of default constructors.

DATE

4(c) "Java string is called immutable" – explain with examples.

Answer: The string is immutable in Java because of the security synchronization and concurrency, caching and class loading. The reason of making string final is to destroy the immutability and to not allow others to extend it. The string objects are cached in the string pool, and it makes the string immutable.

NOTES

7(a) What is multithreaded programming? What are the advantages of multithreading?

Answer: Multithreading is a model of program execution that allows for multiple threads to be created within a process, executing independently but concurrently sharing process resources. Depending on the hardware, threads can run fully parallel if they are distributed to their own CPU core.

The primary function of multithreading is to simultaneously run or execute multiple tasks. These tasks are represented as threads in a Java program and have a separate execution path. Also,

DATE

handling of multithreaded Java programs is easy because we can decide the sequence in which execution of Java threads take place.

Multithreading allows the execution of multiple parts of a program at the same time. These parts are known as threads and are lightweight processes available within the process. So, multithreading leads to maximum utilization of the CPU by multitasking.

Fall 2016

1(a)

The class is at the core of Java. It is the logical concept upon which the entire Java language is built. because it defines the shape and the nature of an object. As such, the class forms the basic for object-oriented programming in Java.

A class creates a new data type that can be used to create objects. A class creates a logical framework which defines the relationship between its members. When we declare an object of a class, we are creating an instance of that class. Therefore, a class is a logical construct. An object

DATE

occuples space in memory.

Examples

1(b) Call by value means calling a method with a parameter as value. Through this, the argument value is passed to the parameter. While Call by Reference means calling a method with a parameter as a reference. Through this, the argument reference is passed to the parameter.

There is only call by value in java, not call by reference. If The changes being done in the called method, is not affected in the calling ~~met~~ method.

In case of call by reference or original value is changed if we made changes in the called method. If we pass object in place

DATE

of any primitive values, original value will be changed.