PAULA MANESE
CFG-NANODEGREE
FULL-STACK 1
MR. SCOTT ADAMS
AUGUST 5, 2022

HOMEWORK WEEK 3

This week's homework is a research based one. You'll need to conduct independent learning, in combination with existing material (where available), to answer the questions below.

The reason for this homework is to ensure you are aware of critical topics in CS.

These topics were difficult to cover within the existing lesson schedules, but due to their importance are placed within the homework instead. Make sure to research, learn and then answer the following:

1. **What is OOP? How may you have already made use of it (e.g. class components)?**
    a. ***Feel free to give a fairly light answer here - as you'll need to do the deep-part / actual meat in the following questions when you cover each of OOP's pillars.***

ANSWER:

OBJECT ORIENTED PROGRAMMING

- is a methodology or paradigm to design a program using classes and objects. It simplifies the software development and maintenance by providing some concepts such as Objects, Class, Polymorphism, Abstraction, etc.

OBJECT

-Any entity that has state and behavior is known as an object. For example a chair, pen,table, keyboard, bike, etc. It can be physical or logical. An Object can be defined as an instance of a class. An object contains an address and takes up some space in memory. Objects can communicate without knowing the details of each other's data or code. The only necessary thing is the type of message accepted and the type of response returned by the objects.

Example:

A dog is an object because it has states like color, name, breed, etc. as well as behaviors like wagging the tail, barking, eating, etc.

CLASS

A **class** are templates that are used to create objects, and to **define** object data types and methods. Core properties include the data types and methods that may be used by the object

A class can also be defined as a blueprint from which you can create an individual object. Class doesn't consume any space.

2. **What is Polymorphism?**

ANSWER:

- is the ability of an object to take on many forms. The most common use of polymorphism in OOP occurs when a parent class reference is used to refer to a child class object.

They can have properties and methods. For example, we could create an object defined as class Car. This car could have:

Properties, such as: color, make, model, current speed
Methods, which are functions invoked by the class, such as: go, stop, park, turn left, turn right

For the above Car object to be polymorphic, its properties and methods can be called using the same class name "Car" but invoke different types depending on how it is used in the code.

Types of Polymorphism

There are four main types of polymorphism:

Subtype polymorphism (Runtime)

Subtype polymorphism is the most common kind of polymorphism. It is usually the one referenced when someone says, "The object is polymorphic."

A subtype polymorphism uses one class name to reference multiple kinds of subtypes at once. In the car example above, we create the object class, "Car", and define multiple subtypes of it to be: Ford, Chevrolet, Kia, Nissan, Volkswagen, BMW, Audi, and Tesla. Each car has its property color.

All subtypes can be referenced interchangeably by using class Car.

Every car has a property, color, and now we can get the color from each subtype of the Car class.

The polymorphism takes place at runtime. We can write a function to fetch the color of any one of the car subtypes. The function may look like this (not written in any specific programming language):

```
getColor(BMW)
```

→ returns red

```
getColor(Audi)
```

→ returns blue

```
getColor(Kia)
```

→ returns yellow

Parametric polymorphism (Overloading)

A parametric polymorphism specifically provides a way to use one function (the same code) to interact with multiple types.

An example would be a list of types. A parametric polymorphism could remove, add, or replace elements of this list regardless of the element's type.

The code, which could be a parametric polymorphism function written in Python:

```
for (element in list):

list.remove(element)
```

Ad hoc polymorphism (Compile-time)

For ad hoc polymorphism, functions with the same name act differently for different types.

Python is dynamically typed (it does not require the type to be specified). The Python addition function is an ad hoc polymorphism because it is a single function of the same name, "+", that works on multiple types.

Both (Python):

1.

```
3 + 4
```

```
"Foo" + "bar"
```

2.

```
→ 7
```

```
→ "Foobar"
```

For values of type int, the addition function adds the two values together, so 3+4 returns 7. For values of type String, the addition function concatenates the two strings, so “Foo” + “bar” returns, “Foobar”.

Coercion polymorphism (Casting)

Coercion polymorphism is the direct transformation of one type into another. It happens when one type gets cast into another type. Before, polymorphism occurred through interacting with different types through the object class or the functions. An object’s type could be selected when the program was run. A single function could work with different types.

A simple example is transforming number values from ints, to doubles, to floats, and vice versa. Depending on the programming language, either the type can be defined on the variable, or sometimes there exists a method on the types to convert their value to another type.

1. Python will convert types like this:

```
int(43.2) #Converts a double to an int
```

```
float(45) #Converts an int to a float
```

2. Java might convert the type simply by defining the type:

```
int num = 23;
```

```
double dnum = num;
```

3. **What is Abstraction?**

ANSWER:

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

Another way, it shows only essential things to the user and hides the internal details, for example, sending SMS where you type the text and send the message. You don't know the internal processing about the message delivery.

Abstraction lets you focus on what the object does instead of how it does it.

There are two ways to achieve abstraction:

-Abstract class (0 to 100%)

-Interface (100%)

Abstract Class

A class which is declared as abstract is known as an **abstract class**. It can have abstract and non-abstract methods. It needs to be extended and its method implemented. It cannot be instantiated.

Some points to remember :

- An abstract class must be declared with an abstract keyword.
- It can have abstract and non-abstract methods.
- It cannot be instantiated.
- It can have constructors and static methods also.
- It can have final methods which will force the subclass not to change the body of the method.

Abstract Method

A method which is declared as abstract and does not have implementation is known as an abstract method.

4. **What is Inheritance?**

ANSWER:

-is a mechanism in which one object acquires all the properties and behaviors of a parent object. It is an important part of OOPs (Object Oriented programming system).

-The idea behind inheritance is that you can create new classes that are built upon existing classes. When you inherit from an existing class, you can reuse methods and fields of the parent class. Moreover, you can add new methods and fields in your current class also.

-Inheritance represents the **IS-A relationship** which is also known as a parent-child relationship.

Advantages of Inheritance

- For Method Overriding (so runtime polymorphism can be achieved).
- For Code Reusability.

Terms used in Inheritance

- **Class:** A class is a group of objects which have common properties. It is a template or blueprint from which objects are created.
- **Sub Class/Child Class:** Subclass is a class which inherits the other class. It is also called a derived class, extended class, or child class.
- **Super Class/Parent Class:** Superclass is the class from where a subclass inherits the features. It is also called a base class or a parent class.
- **Reusability:** As the name specifies, reusability is a mechanism which facilitates you to reuse the fields and methods of the existing class when you create a new class. You can use the same fields and methods already defined in the previous class.

5. **What is encapsulation?**

ANSWER:

- ensures that each object in your code should control its own state. Encapsulation is defined as the wrapping up of data under a single unit. It binds together code and the data it manipulates.
- It is a technique of restricting a user from directly modifying the data members or variables of a class in order to maintain the integrity of the data. How do we do that? We restrict the access of the variables by switching the access-modifier to private and exposing public methods that we can use to access the data.
- may also mean restricting direct access to certain components of an object so that users can't access the state values for all variables of a particular object. Therefore, encapsulation can be used to hide data members and functions associated with an instantiated class or object. Encapsulation is analogous to a capsule where the mixture of medicines inside the pill represents the data and methods while the hard outer shell could be thought of as the class.

6. **What is:**
   a. **Agile development?**

ANSWER:

-is a type of methodology and it is a type of software development project management.

-practices include requirements discovery and solutions improvement through the collaborative effort of self-organizing and cross-functional teams with their customer(s)/end user(s), adaptive planning, evolutionary development, early delivery, continual improvement, and flexible responses to changes in requirements, capacity, and understanding of the problems to be solved.

   b. **Waterfall development?**

ANSWER:

-is a breakdown of project activities into linear sequential phases, where each phase depends on the deliverables of the previous one and corresponds to a specialization of tasks. The approach is typical for certain areas of engineering design.

- In software development, it tends to be among the less iterative and flexible approaches, as progress flows in largely one direction ("downwards" like a waterfall) through the phases of conception, initiation, analysis, design, construction, testing, deployment and maintenance.The waterfall development model originated in the manufacturing and construction industries,where the highly structured physical environments meant that design changes became prohibitively expensive much sooner in the development process. When first adopted for software development, there were no recognised alternatives for knowledge-based creative work

c. How do they differ? Which is suited for which situation?

ANSWER:

Waterfall vs Agile Key Difference

- Waterfall is a Liner Sequential Life Cycle Model whereas Agile is a continuous iteration of development and testing in the software development process.
- In Agile vs Waterfall difference, the Agile methodology is known for its flexibility whereas Waterfall is a structured software development methodology.
- Comparing the Waterfall methodology vs Agile which follows an incremental approach whereas the Waterfall is a sequential design process.
- Agile performs testing concurrently with software development whereas in Waterfall methodology testing comes after the "Build" phase.
- Agile allows changes in project development requirement whereas Waterfall has no scope of changing the requirements once the project development starts.

Agile	Waterfall
It separates the project development lifecycle into sprints.	Software development process is divided into distinct phases.
It follows an incremental approach	Waterfall methodology is a sequential design process.
Agile methodology is known for its flexibility.	Waterfall is a structured software development methodology so most times it can be quite rigid.
Agile can be considered as a collection of many different projects.	Software development will be completed as one single project.
Agile is quite a flexible method which allows changes to be made in the project development requirements even if the initial planning has been completed.	There is no scope of changing the requirements once the project development starts.
Agile methodology, follow an iterative development approach because of this planning, development, prototyping and other software development phases may appear more than once.	All the project development phases like designing, development, testing, etc. are completed once in the Waterfall model.
Test plan is reviewed after each sprint	The test plan is rarely discussed during the test phase.
Agile development is a process in which the requirements are expected to change and evolve.	The method is ideal for projects which have definite requirements and changes not at all expected.
In Agile methodology, testing is performed concurrently with software development.	In this methodology, the "Testing" phase comes after the "Build" phase
Agile introduces a product mindset where the software product satisfies needs of its end customers and changes itself as per the customer's demands.	This model shows a project mindset and places its focus completely on accomplishing the project.

Agile methdology works exceptionally well with Time & Materials or non-fixed funding. It may increase stress in fixed-price scenarios.	Reduces risk in the firm fixed price contracts by getting risk agreement at the beginning of the process.
Prefers small but dedicated teams with a high degree of coordination and synchronization.	Team coordination/synchronization is very limited.
Products owner with team prepares requirements just about every day during a project.	Business analysis prepares requirements before the beginning of the project.
Test team can take part in the requirements change without problems.	It is difficult for the test to initiate any change in requirements.
Description of project details can be altered anytime during the SDLC process.	Detail description needs to implement waterfall software development approach.
The Agile Team members are interchangeable, as a result, they work faster. There is also no need for project managers because the projects are managed by the entire team	In the waterfall method, the process is always straightforward so, project manager plays an essential role during every stage of SDLC.

Once complete, please return to your instructor your answers! Remember:

- Justify and be critical of everything! This distinguishes a great answer from a good answer
- Analyse - why does x even exist? Who needs it or uses it? Who is it important to, what's the point of it at all?