IHUOMA FAVOUR AGBARU

HOMEWORK WEEK 3

1. What is OOP? How may you have already made use of it (e.g. class components)?
   1. *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 can be defined as a computer programming paradigm that arranges the architecture of software around data or objects rather than functions and logic. An object is a data field with particular characteristics and behavior.

The organization of an object-oriented program also makes the method beneficial to collaborative development, where projects are divided into groups. Additional benefits of OOP include code re-usability, scalability and efficiency.

**Structure Of Object-Oriented Programming**

The building blocks of object-oriented programming include the following:

* **Classes:** These are user-defined data types that act as the blueprint for individual objects, attributes and methods.
* **Objects:** They are instances of a class created with specifically defined data. Objects can correspond to real-world objects or an abstract entity. When class is defined initially, the description is the only object that is defined.
* **Methods:** These are functions that are defined inside a class that describe the behaviors of an object. Each method contained in class definitions starts with a reference to an instance object. Additionally, the subroutines contained in an object are called instance methods. Programmers use methods for re-usability or keeping functionality encapsulated inside one object at a time.
* **Attributes:** They are defined in the class template and represent the state of an object. Objects will have data stored in the attributes field. Class attributes belong to the class itself.

**BENEFITS OF OBJECT ORIENTED PROGRAMMING**

* Modularity: Encapsulation enables objects to be self-contained, making troubleshooting and collaborative development easier.
* Reusability:  Code can be reused through inheritance, meaning a team does not have to write the same code multiple times.
* Productivity: Programmers can construct new programs quicker through the use of multiple libraries and reusable code.
* Easily upgradable and scalable: Programmers can implement system functionalities independently.
* Interface descriptions: Descriptions of external systems are simple, due to message passing techniques that are used for objects communication.
* Security: Using encapsulation and abstraction, complex code is hidden, software maintenance is easier and [internet protocols](https://www.techtarget.com/searchunifiedcommunications/definition/Internet-Protocol) are protected.
* Flexibility: Polymorphism enables a single function to adapt to the class it is placed in. Different objects can also pass through the same interface.

**APPLICATIONS OF OBJECT ORIENTED PROGRAMMING**

* Computer graphics applications
* Object-oriented database
* User-interface design
* Real-time systems
* Simulation and modeling
* Client-Server System
* Artificial Intelligence System
* Computer- aided design and Computer- aided Manufacturing (CAD/CAM)Software.
* Office automation system.

A program to show OOP

class Car extends React.Component {

  constructor(props) {

      super(props);

   this.state = {

      make: "Toyota",

      model: "2016",

      color: "gray"

      };

      }

1. What is Polymorphism?

**ANSWER:** Polymorphism is an object-oriented programming concept that refers to the ability of a variable, function, or object to take on multiple forms. Objects can have multiple forms and are intended to have shared behaviour. There will be less need for duplicating code because the program will determine which usage or meaning is required for each execution of that object from a parent class. The functionality of the parent class is then expanded by a child class that is formed. Different sorts of objects can pass via the same interface thanks to polymorphism.

**TYPES OF POLYMORPHISM**

* Compile time polymorphism and
* Runtime polymorphism.

**ADVANTAGES OF POLYMORPHISM:**

* It helps software developers reuse code and classes once written, tested, and implemented.
* A single variable name can be used to store variables of multiple data types (float, double, long, int, etc).
* It helps compose powerful, complex abstractions from simpler ones.

1. What is Abstraction?

**ANSWER:** Abstraction in Object-Oriented Programming means showing only the essential attributes of something and concealing any implementation information that is unnecessary to the user.Objects conceal any extraneous implementation code and only expose internal mechanisms that are necessary for the use of other objects. The functionality of the derived class can be increased. This idea can make it simpler for programmers to introduce new features or changes in the future.

**ADVANTGES OF ABSTRACTION**

* Abstraction allows developers to avoid duplication of code in a program and increase re-usability.
* With abstraction, an application security is increased as developers are only allowed access to details important to them.

1. What is Inheritance?

**ANSWER:** The process through which one class acquires the attributes and methods of another class is known as inheritance.Code from other classes may be reused by classes. The ability to assign relationships and subclasses allows developers to reuse similar functionality while yet retaining a distinct hierarchy. This attribute of OOP speeds up development and provides more accuracy by requiring a more in-depth investigation of the data.

**FORMS OF INHERITANCE IN OOP**

* **Single inheritance**

This is a form of inheritance in which a class inherits only one parent class.

* **Multiple inheritance**

Multiple inheritances when a class inherits more than one parent class.

* **Multi-level Inheritance**

This is when a class is inherited by another class and that said class gets inherited by another class and the list continues.(class 1 is inherited by class 2 and class 2 is inherited by class 3…..)

* **Hierarchical inheritance**

In hierarchical inheritance various Child classes inherit a single Parent class.

* **Hybrid Inheritance**

This is When there is a combination of more than one form of inheritance.

1. What is encapsulation?

**ANSWER:**  encapsulation can be defined as the idea of bundling data and methods that work on that data within one unit. It is also about wrapping data and methods into a single class and protecting it from outside.

**TYPES OF ENCAPSULATION IN OOPs**

* **Data Member Encapsulation:** This is defined as private members of the class.
* **Method Encapsulation:** This is hiding methods used for internal implementation that does not need to be visible to the public.
* **Class Encapsulation:** We encapsulate the class by defining it as private and hiding it from user access.

**ADVANTAGES OF ENCAPSULATION**

* Increase code reusability.
* Increase Robustness.
* Better code maintenance.
* Data security and information hiding.
* Improves code clarity and comprehension.

1. What is:
   1. Agile development?

**ANSWER:** Agile development is a process that allows **continuous repetition/iteration** of development and testing throughout the software development lifecycle of the project.

**The agile software development has four core values namely:**

* Individual and team interactions over processes and tools.
* Working software over comprehensive documentation.
* Customer collaboration over contract negotiation.
* Responding to change over following a plan.

**The following are the Agile processes for software development.**

* **SCRUM: This is a process** which concentrates specifically on how to manage tasks within a team-based development environment.
* **EXTREME PROGRAMMING:** This process is helpful when there is constantly changing demands or requirements from the customers or when they are not sure about the functionality of the system.
* **CRYSTAL METHODOLOGIES:** This is based on three concepts. (a) Chartering (b) Cyclic delivery and (c)Wrap-up.
* **DYNAMIC SOFTWARE DEVELOPMENT:** This is is a Rapid Application Development approach to software development and provides an agile project delivery framework.
* **FEATURE DRIVEN DEVELOPMENT:** This is focused around “designing & building” features.
* **LEAN SOFTWARE DEVELOPMENT:** This is aimed at increasing speed of software development and decreasing cost.
  1. Waterfall development?

**ANSWER:** This is defined as a system development life cycle used to create a system with a linear and sequential approach I.e from one phase to another phase approach.

**SEQUENTIAL PHASES INVOLVED IN WATERFALL MODEL**

* Requirement gathering
* System Analysis
* System Design
* Implementation development of software
* Integration and testing
* Deployment of system
* System maintenance.

**Advantages of Waterfall Model**

* Easy to use
* Easy to manage as each phase has specific outputs and review process.
* Clearly-defined stage.
* Works well for smaller projects where requirements are very clear
* Process and output of each phase are clearly mentioned in the document.  
    
  **Disadvantages of Waterfall Model**
* It doesn’t allow much reflection or revision. When the product is in testing phase, it is very difficult to go back and change something which is left during the requirement analysis phase.
* Risk and uncertainty are high.  
  1. How do they differ? Which is suited for which situation?

**DIFFERENCES BETWEEN AGILE METHOD AND WATERFALL METHOD**

|  |  |  |
| --- | --- | --- |
| **S/N** | **AGILE METHOD** | **WATERFALL METHOD** |
| **1** | The Agile process in software engineering is broken into individual models that designers work on. | The design process is not broken into an individual models |
| 2 | The customer has early and frequent opportunities to look at the product and make decision and changes to the project | The customer can only see the product at the end of the project |
| 3 | Agile model is considered unstructured compared to the waterfall model. | Waterfall model are more secure because they are so plan oriented. |
| 4 | Small projects can be implemented very quickly. For large projects, it is difficult to estimate the development time. | All sorts of project can be estimated and completed. |
| 5 | Development process is iterative, and the project is executed in short (2-4) weeks iterations. Planning is very less. | The development process is phased, and the phase is much bigger than iteration. Every phase ends with the detailed description of the next phase. |
| 6 | Every iteration has its own testing phase. It allows implementing regression testing every time new functions or logic are released. | Only after the development phase, the testing phase is executed because separate parts are not fully functional. |
| 7 | At the end of every sprint, user acceptance is performed | User acceptance is ****performed**** at the end of the project. |
| 8 | Testers and developers work together. | Testers and developers work separately. |
| 9 | It requires close communication with developers and together analyze requirements and planning. | Developer does not involve in requirement and planning process. Usually, time delays between tests and coding. |