**Quiz-5(V1)**

class Logger {

private:

static Logger\* instance;

int data;

Logger(int data) {

this->data = data;

}

public:

static Logger\* getInstance() {

//missing implementation

}

};

1. Identify which design pattern is followed by the **class Logger. (2)**

Singleton

1. Name the **category** to which the design pattern belongs, that you mentioned in **part 1**. **(1)**

Creational / object creational

1. Write the implementation of the function **getInstance(). (3)**

**static Logger\* getInstance() {**

**if (instance == nullptr) {**

**instance = new Logger(1); // Assuming 1 as the default con return instance;number**

**}**

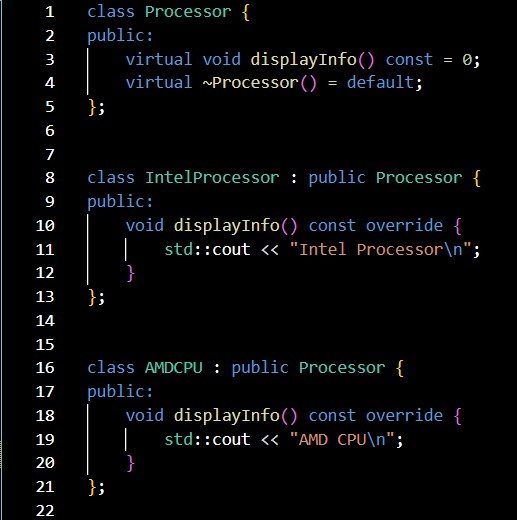
1. What is the purpose of the method **getInstance(). (2)**

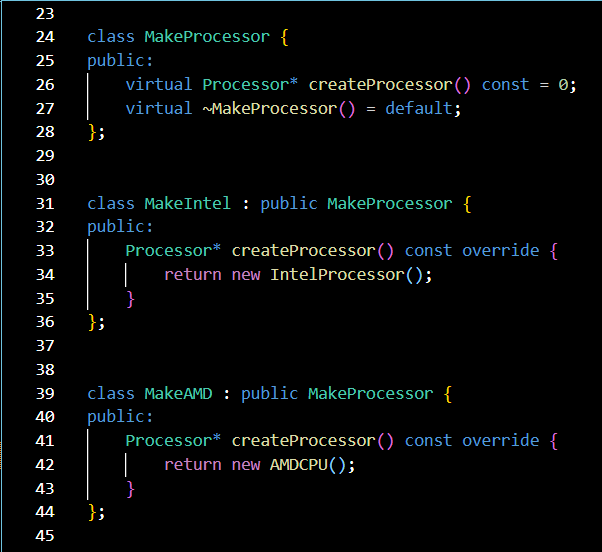
It returns the only instance of the class and instantiates if there is none.

1. Why is the private data member **Logger\*** **instance** declared as **static**? **(2)**

Static variables are shared across all the objects of the classes. Moreover, it behaves as a global point of access.

**Quiz-5(V2)**

****



**The given code follows the Factory Design Pattern.**

1. Name the **category/type** to which the Factory Design pattern belongs. **(1)**

Creational / object creational

1. Identify the **base factory class. (2)**

MakeProcessor

1. Identify the **base product class. (2)**

Processor

1. Identify the **Factory Method. (2)**

Create processor()

1. The programmer decides to add one more product. Write the implementation of that product named **‘productX’. (3)**

**Class** productX : **public** Processor {

Void displayinfor() const override{

Cout << “productX” << endl;

}

}