Question 1: Write the outputs of following code snippets. Each question carries 3 marks

Marks [12]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| class base{ private: int a; protected: int b; public:  void fun(){ cout<<"this works t'•  class child: private base{ public:  void funl(){ base::fun(); base::b = 0;  int main(){ child cl; cl.fun(); | | | class TableofContents { private:  list<string> items;  public:  TableofContents()  cout<<" Table of Content is shown\n ;  void addltem(string item) { items.push\_back(item);  class Book { public:  TableofContents toc; list<string> sections; list<string> chapters;  Book() { toc = TableofContents();  int main() {  Book bookl = Book(); getchar(); return 0; | | |
| OutPut:  ERROR | | | OutPut:  Ouput:  Table of Content is shown\n ; | |  |
|
|  | | | | |
| class A{ int a; public:  A(int i){  void assign(int i){    int return\_value(){ return a;  int main(int argc, char const \*argv[l)  A obj; obj .assign(5); cout<<obj.return value();   1. 55 2. Error 3. Segmentation Fault | class A{ int a; public:  constructor called" ;  class B{ static A a; public:  constructor called",  static A get(){ return a;  int main(int argc, char const \*argv[l)    A al = b.get();  A a2 = b.get();  A a3 = b.get();  Note: In the following C++ code how many times the string "A's constructor called" will be rinted? | |
| OutPut: (c) ERROR | OutPut: ONE TIME | |

Question 2: Give the short answer to the following questions. Marks[61

I. What is the purpose of access modifiers in OOP languages? [21

Answer: There most common purpose access modifiers are in OOP language:

Public: This modifiers makes the class memeber accessible to any other class.

Private: This modifier makes the class members accessible only within the class in which they are defined.

Protected: This modifier makes the class members accessible within the class and any subclasses of that class.

11. If we want to access the private members of a class in the child class what do we need to change? [011

Answer:

We will change private member into public memeber. Because its accessible in

the main.

Ill. Determine the accessibility of functions and data members in the following scenarios. 131

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | Scenario | Accessible | Not Accessible | | A private data member is declared in a class in accessible by its object in the main function. |  |  | | A Protected function defined in parent class by the functions of the child class. |  |  | | A public data member of the parent class by the object of child class |  |  | |  |

Question 3: There are FIVE (5) Errors (Syntax & Logical) in code given below. Identify error lines and write correct code. Marks 1101

|  |  |
| --- | --- |
| 1. class Bl { 2. public: 3. i; 4. int j; 5. void g(int) { } 6. class B2 { 7. public: 8. mt J 9. void g() { }        1. class D : public Bl; class public B2 2. public: 3. mt 1 15. 4. int main() { 5. D dobj; 6. D \*dptr = &dobj; 7. dptr->i = 5; 8. dptr->j = 10; 9. dobj.g();   22- } | Error#l |
| Line # 3 |
| Correct line of code:  Int I; |
| Error#2 |
| Line # 10 |
| Correct line of code:  VOID G(INT) |
| Error#3 Line # 12 |
|
| Correct line of code:  class D : public B1, public B2 |
| Error#4 |
| Line # 17 |
| Correct line of code:  D d; |
| Error#5 |
| Line # 21 |
| Correct line of code:  d.g(); |
|  |

Question#4:

You have to develop a game that has multiple characters. These characters share some common properties like id, name, maximum power and strength. There are other properties as well that they have their own like Doremon has properties like a list of names gadgets and the name of partner, Benten has the watch Name, a list of names powers and total charge of the watch. There are also some common actions that they can perform like walk, jump and eat. Doremon can show Gadgets, launch attack and fly . Benten can perform the actions like rotate watch, fight and drive.

Implement the game using Inheritance in C++. Marks [07]

#include <iostream>

#include <string>

#include <vector>

using namespace std;

// Base class for all characters

class Character {

protected:

int id;

string name;

int maxPower;

int strength;

public:

Character(int id, string name, int maxPower, int strength)

: id(id), name(name), maxPower(maxPower), strength(strength) {}

void walk() {

cout << name << " is walking.\n";

}

void jump() {

cout << name << " is jumping.\n";

}

void eat() {

cout << name << " is eating.\n";

}

};

// Doremon class derived from Character class

class Doremon : public Character {

private:

vector<string> gadgets;

string partner;

public:

Doremon(int id, string name, int maxPower, int strength,

vector<string> gadgets, string partner)

: Character(id, name, maxPower, strength), gadgets(gadgets), partner(partner) {}

void showGadgets() {

cout << name << "'s gadgets: ";

for (auto gadget : gadgets) {

cout << gadget << ", ";

}

cout << "\n";

}

void launchAttack() {

cout << name << " is launching an attack!\n";

}

void fly() {

cout << name << " is flying.\n";

}

};

// Benten class derived from Character class

class Benten : public Character {

private:

string watchName;

vector<string> powers;

int watchCharge;

public:

Benten(int id, string name, int maxPower, int strength,

string watchName, vector<string> powers, int watchCharge)

: Character(id, name, maxPower, strength), watchName(watchName), powers(powers), watchCharge(watchCharge) {}

void rotateWatch() {

cout << name << " is rotating the watch.\n";

}

void fight() {

cout << name << " is fighting.\n";

}

void drive() {

cout << name << " is driving.\n";

}

};

int main() {

// Create a Doremon object

vector<string> doremonGadgets = {"Take-copter", "Anywhere Door", "Bamboo-Copter", "Small Light"};

Doremon doremon(1, "Doremon", 100, 50, doremonGadgets, "Nobita");

// Create a Benten object

vector<string> bentenPowers = {"Fireball", "Freeze", "Big Chill", "Waterblast"};

Benten benten(2, "Benten", 200, 70, "Omnitrix", bentenPowers, 80);

// Use the common actions

doremon.walk();

benten.jump();

// Use the specific actions for each character

doremon.showGadgets();

benten.rotateWatch();

// Use the common actions again

doremon.eat();

benten.walk();

return 0;

}