Quiz 19 Section B

OOP Section-B Quiz
Note: There might be MCQs with more than 1 option correct you need to mark the option which you think is closest one.

The respondent's email (**bsse23002@itu.edu.pk**) was recorded on submission of this form.

Total types of constructors in C++ are? * 1 point

- 1
- 2
- 3
- 4

What would be output of following code snippet? * 1 point

```
class Base {
public:
    void print() { cout << "Base"; }
};
class Derived : public Base {
public:
   void print() { cout << "Derived"; }
};
int main() {
   Base *ptr = new Derived();
   ptr->print();
   delete ptr;
   return 0;
}
```

- ○ Base
- ◉ Derived
- ○ Error
- ○ No Output

Which feature of OOP allows the same function to behave differently in different contexts? * 1 point

- ○ Inheritence
- ○ Abstraction
- ○ Encapsulation
- ◉ Non of these

Consider the following code snippet. What is the most likely reason for making the display function const? * 1 point

```
class Sample {
  public:
  void display() const { // Display logic here }
}
```

- ○ To allow the function to modify the member variables of the class.
- ◉ To prevent the function from modifying any member variables of the class.
- ○ To enable the function to be called on non-const objects.
- ○ Non of above

Which of the following access specifiers will make member variables accessible only within the class and its friends? * 1 point

- ○ public
- ○ protected
- ○ friend
- ◉ Non of above

In the context of encapsulation and data hiding in C++, which of the following statements are correct? * 1 point

- [ ] Encapsulation involves bundling the data and methods that operate on the data within one unit.
- [ ] Data hiding is a technique that restricts direct access to an object's data.
- [x] Public member functions provide controlled access to the hidden data of an object.
- [ ] All member variables should be made public to simplify access from outside the class.

*

1 point

Which of the following is not a type of inheritance?

- [ ] Multiple
- [ ] Multilevel
- [ ] Hierarchical
- [x] Non of Above

How does C++ resolve the diamond problem? * 1 point

- [ ] By using the override keyword to specify which base class method to use.
- [ ] By automatically detecting and removing duplicate inheritance paths.
- [ ] C++ does not provide a built-in way to resolve the diamond problem; it must be manually managed by the programmer.
- [x] Non of above

Which of the following statements is true regarding virtual inheritance? * 1 point

- ○ Virtual inheritance significantly improves the performance of the derived class by optimizing memory usage.
- ◉ Virtual inheritance is a mechanism that ensures multiple copies of a base class are not created when derived through multiple paths.
- ○ When using virtual inheritance, the constructor of the virtual base class is called by the most derived class.
- ○ Virtual inheritance allows for multiple instances of the same base class to coexist within a derived class.

When dealing with inheritance in C++, which of the following statements are correct? * 1 point

- ○ Protected members of the base class are accessible in the derived class but not outside of it
- ○ Private members of the base class are inaccessible directly from the derived class.
- ◉ Multiple inheritance can lead to the diamond problem, which can be resolved using virtual inheritance.
- ○ Inheritance allows for code reuse but can make code more difficult to understand.