

1- State/Write whether each of the following is True or False. (Every two wrongs will take away one right.) (1 pt. each):

- a. ~~False~~ Any C++ program that prints three lines of output must contain three output statements using *cout*.
- b. ~~True~~ C++ consider the variables *numero* and *Numero* to be identical.
- c. ~~True~~ Static data members of a class have class scope.
- d. ~~False~~ An array can store many different types of values.
- e. ~~False~~ If *y* and *z* are declared as: *int y, z;* then I can write *y = &z;*
- f. ~~False~~ The name of an array is a pointer to the last element of the array.
- g. ~~False~~ Class *destructors* can take arguments.
- h. ~~False~~ A *const* member function of a class can modify *const* objects.
- i. ~~False~~ Constructors cannot specify default arguments.
- j. ~~False~~ A friend function of a class cannot access the protected members of the class.
- k. ~~True~~ A subclass can access the private members of its parent class.
- l. ~~True~~ The precedence of an operator cannot be changed by overloading.
- m. ~~False~~ If the *+* operator is overloaded to work on arrays of integers, then the *+=* operator will be automatically overloaded to work with arrays of integers.
- n. ~~True~~ Operator overloading can be used to create new non-existing operators.
- o. ~~True~~ When deriving a class from a public base class, all members of the base class become public members of the derived class.

9

Find/circle the *syntax* error(s) in each of the following and *correct it/them or explain how to correct it:* (1 pt for each corrected error)

a. `int A2D[2][2] = {{1, 2}, {3, 4}}; A2D[1, 1] = 50;`

`int A2D[2][2] = {{1, 2}, {3, 4}};`

Cannot

~~access~~ write A2D[i][j]
but A2D[i][j] = 50

2

b. `class hour {
public:`

`hour(int);`

`void ~hour();`

`private:`

`int H = 0;`

`};`

`~hour(); // destructor`

1

c. Class easytricks{

public :

~~void~~ Easytricks(int y = 5) {x = y} // constructor

int incrementX () const {return ++x;}

-easytricks (void);

private:

int x; }

(2)

int incrementX() const cannot return ++x;
everytime called it has to increment x
so ~~const~~ ^{therefore} should so it is not const.
→ int incrementX() { return ++x; }

3- What is the main benefit of using inheritance?

The use of member functions of the base class in the derived class, which is the software reusability concept (1 pt)

4- What is the difference between a static variable and an automatic variable? (2 pts)

A static variable keeps value between function calls whereas an automatic variable is always the same when called again

5- Declare x as an integer variable, initialize x to 10, declare y as a pointer to an integer, assign y the address of x, and then write a statement to print the content of x using the dereferencing operator. (2 pts)

```
int x = 10;  
int *y;
```

```
y = &x;
```

cout << "the content of x using the dereferencing operator is" << *y << endl;

- 6- Define a class **person** with three public data members: *first-name*, *last-name*, and *age*. Their default values are: *Mohamad*, *Ladan*, and *35*. Derive a class **student** from class **person**, the class **student** should have one public data member *major*. Overload the stream insertion operator `<<` (*cout*) so that it will print the *first-name*, *last-name*, *age*, and *major* for any student object.

Write the class **person** and its subclass **student**, and a **driver program** that will instantiate an object of a class **student** initialized with your own data (your *firstname*, *lastname*, *age*, and *major*) using the class constructor, and print these data using the overloaded stream insertion operator `<<` (10 pts)

```
class person {  
    public:
```



```
class person {
```

```
public:
```

```
    person (char first-name, char last-name, int = 35); // default
```

```
    char first-name [3];
```

```
    char last-name [3];
```

```
    int age;
```

```
};
```

```
person::person (char first, char last, int old)
```

```
{
    first-name = first; // ends at space
```

```
    last-name = last;
```

```
    age = old;
}
```

```
}
```

①


```

class student : public person
{
friend ostream &operator << (&);
public:

```

3

```

char major;
student (char first, char last, int age, char major)
& person (char first, char last, int age)
{
major = maj;
}
}

```

2

```

student;
ostream &operator << (ostream output)
{
output << first-name << ' ' << last-name << ' ' << age
<< ' ' << major << endl;
return output;
}

```

4


```
#include <string.h>
int main()
{
    student s("Georges", "Bali", 22, "csc");
    cout <<< s;
} // end of driver
```