

Programmation Orientée Object en C++

1. Définir et illustrer à l'aide d'un exemple les mots suivant : (1.5+1.5+1.5=4.5points)
 - Classe
 - Objet
 - Héritage
2. Donner la signification des mots clés suivant : (1.5 point)
 - Virtual
 - Public
 - Private
3. Les constructeurs peuvent-ils être surchargés? (1 point)
 - a- Oui
 - b- Non
4. Quelle est la syntaxe correcte pour inclure un fichier d'en-tête défini par l'utilisateur en C++? (1 points)
 - A- #include <myfile.h>
 - B- #include <myfile>
 - C- #include "myfile"
 - D- #include [myfile]
5. Quelle est la sortie du code suivant? (1.5 points)

```
#include <iostream>
using namespace std;
class calculer {
int x, y;
public:
void val(int, int);
int somme() {
return (x + y);
}
};
void calculer::val(int a, int b) {
x = a;
y = b;
}
int main() {
calculer calculer;
calculer.val(5, 10);
cout << "La somme = " << calculer.somme();
return 0;
}
```

- A- La somme = 5
- B- La somme = 10
- C- La somme = 15
- D- Erreur parce que calculer est utilisé comme nom de classe et nom de variable dans la ligne 19.

6. Quel constructeur de classe est appelé dans le code C++ suivant? (1.5 point)

```
#include<iostream>
using namespace std;
class A {
public:
A()
{ cout << " Le constructeur de la classe A est appelé."
<< endl; }
};
class B {
public:
B()
{ cout << " Le constructeur de la classe B est appelé."
<< endl; }
};
class C: public A, public B {
public:
C()
{ cout << " Le constructeur de la classe C est appelé."
<< endl; }
};
int main()
{
C c;
return 0;
}
```

- A- Classe C
- B- Classe A et B
- C- Classe A, B et C
- D- Erreur de compilation.

7. Quelle est la sortie du code suivant? (1.5point)

```
#include<iostream>
using namespace std;
class A {
private:
int val;
public:
A(int v = 0) : val(v) {}
void display() { cout << "val = " << val << endl;}
};
class B {
private:
int val;
public:
B(int v) : val(v) {}
operator A() const { return A(val); }
};
void f(A a)
{ a.display(); }
```

```
int main() {
    B b(5);
    f(b);
    f(55);
    return 0;
}
```

- A- val = 5 et val = 5
- B- val = 5 et val = 55
- C- val = 5
- D- val = 55

Problème : (8 points)

1. Écrire une classe Complexes permettant de représenter des nombres complexes. Un nombre complexe Z comporte une partie réelle et une partie imaginaire : $Z = \text{PartieRéelle} + \text{PartieImaginaire} * i$
2. Définir à l'aide des propriétés les méthodes d'accès aux attributs de la classe.
3. Définir un **constructeur** par défaut permettant d'initialiser les deux parties du nombre à 0.
4. Définir un **constructeur** d'initialisation pour la classe.
5. Ajouter les méthodes suivantes :
 - **Plus(Complexe)** : Elle permet de retourner le nombre complexe obtenu

en ajoutant au nombre en cours un nombre complexe passé en argument.

- **Moins(Complexe)** : Elle permet de retourner le nombre complexe obtenu en soustrayant au nombre en cours un nombre complexe passé en argument.
- **Afficher ()** : Elle donne une représentation d'un nombre complexe comme suit : $a+b*i$.

6. Écrire un programme permettant de tester la classe Complexe.

Votre programme devra comporté un (ou plusieurs) fichier(s) d'entête , un(des) fichier(s) CPP et un programme principal

Exemple d'exécution :

Nombre Complexe 1:

Donner la partie réelle: 2

Donner la partie imaginaire: 4

Nombre Complexe 1: $2+4i$

Nombre Complexe 2:

Donner la partie réelle: 3

Donner la partie imaginaire: -2

Nombre Complexe 2: $3-2i$

La somme:

$5+2i$

La Différence:

$-1+6i$

conseillée

Une bonne présentation de la copie est