HE2B ESI

Langage C++
Examen de janvier

/160

R. Absil (abs)

12 janvier 2017

Consignes générales

1. Notez votre nom, prénom, groupe et matricule sur chacune des feuilles que vous remettez à votre maître-assistant, y compris les brouillons.
2. Rédigez vos réponses en français correct : soignez votre orthographe et votre grammaire.
3. Justifiez chacune de vos réponses de façon complète et concise : une réponse incorrecte justifiée avec de bons arguments peut rapporter quelques points, alors qu'une bonne réponse non justifiée sera toujours comptée comme nulle.
4. L'examen est à cahier fermé.
5. L'examen dure 2h.

Question 1. Analysez les codes suivants, sachant que toutes les options de compilation nécessaires sont fournies (p. ex., `-std=c++11`, `-fpermissive`, etc.). Pour chacun de ces codes, on souhaite produire sans erreur l'affichage demandé. Le comportement réel du programme est également décrit. /110

- S'il y a une erreur de compilation ou à l'exécution, dites à quelle(s) ligne(s) le cas échéant, pourquoi l'erreur apparaît et que faire pour corriger le programme en modifiant le code fourni de manière minimale. Notez que dans ce cas, les erreurs sont de type sémantique et / ou de logique (ne cherchez donc pas un « `;` » ou un « `std::` » manquant).
- Si le programme s'exécute sans erreurs mais ne produit pas l'affichage souhaité, corrigez le pour que ce soit le cas.
- Il est possible que plusieurs erreurs différentes apparaissent au sein d'un même code.
- Il est possible qu'il ne soit pas possile de corriger le code. Le cas échéant, justifiez pourquoi.
- Quel que soit le comportement du programme, si son comportement dépend de l'architecture du système (compilateur, processeur, etc.), précisez-le.
- Chaque code vaut 10 points, il y a 11 codes.
- Répondez sur le papier ministre, vous pouvez référencer le code via les numéros de ligne.

Nom : Prénom : Groupe : Matricule :

Code 1

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7    unsigned i = 5;//vous ne pouvez pas changer cette ligne
8    while(i >= 0)
9    {
10     if(i != 0)
11       cout << i << endl;
12     i--;
13   }
14   cout << "BOOM" << endl;
15 }
```

Affichage souhaité du code 1

```
1 5
2 4
3 3
4 2
5 1
6 BOOM
```

Affichage produit par le code 1

```
1 Boucle infinie
```

Code 2

```
1  #include <iostream>
2
3  void swap(int * i, int * j)
4  {
5    int * tmp = i;
6    i = j; j = tmp;
7  }
8
9  int main()
10 {
11   int i = 2; int j = 3; swap(&i, &j);
12   cout << i << "␣" << j << endl;
13 }
```

Affichage souhaité du code 2

```
1 3 2
```

Affichage produit par le code 2

```
1 2 3
```

Nom : Prénom : Groupe : Matricule :

Code 3

```
1  #include <iostream>
2  class A {};
3  class B {
4    A * a;
5    public:
6      B() : a(new A()) {}
7      ~B() { delete a; }
8      void print() {  cout << "Talk to the hand" << endl;  }
9  };
10
11 void f(B b) { std::cout << "I lied" << std::endl; }
12
13 int main()
14 {
15   B b;
16   f(b);
17   b.print();
18
19   B bb;
20   bb = b;
21   bb.print();
22
23   bb = bb;
24   bb.print();
25 }
```

Affichage souhaité du code 3

```
1  I lied
2  Talk to the hand
3  Talk to the hand
4  Talk to the hand
```

Affichage produit par le code 3

```
1  Erreur de segmentation
```

Code 4

```
1  #include <iostream>
2  int main() {
3    int i = 5; int j = 2; double d = 1.25; //vous ne pouvez pas modifier cette ligne
4    std::cout << (j / i + d) << std::endl;
5  }
```

Affichage souhaité du code 4

```
1  3.75
```

Affichage produit par le code 4

```
1  3.25
```

Nom : Prénom : Groupe : Matricule :

Code 5

```
1  #include <iostream>
2  #include <list>
3
4  int main()
5  {
6    std::list<int> l;
7    for(int i = 0; i <= 6; i++)
8      l.push_back(i * i - i);
9    for(int i = 0; i < l.size(); i++)
10   {
11     auto j = l.rend();
12     std::cout << *j << " ";
13     l.pop_back();
14   }
15 }
```

Affichage souhaité du code 5

```
1  30 20 12 6 2 0 0
```

Affichage produit par le code 5

```
1  30 20 12 6
```

Code 6

```
1  #include <iostream>
2
3  class A
4  {
5    int i;
6    public:
7      A operator +(int i) { std::cout << "I am one with the force" << std::endl; }
8      A operator +(A a) { std::cout << "and the force is with me" << std::endl; }
9  };
10
11 int main()
12 {
13   int i1; A a1; A a2;
14   i1 + a1 + a2;
15 }
```

Affichage souhaité du code 6

```
1  I am one with the force
2  and the force is with me
```

Affichage produit par le code 6

```
1  Erreur de compilation
```

Nom : Prénom : Groupe : Matricule :

Code 7

```
1  #include <iostream>
2
3  template<class T> class Brol
4  {
5    public:
6      T & t; int & i;
7      Brol(T & t) { this-> t = t; }
8      Brol(int & i) { this-> i = i; }
9      void print() { std::cout << t << " " << i << std::endl;}
10 };
11
12 int main()
13 {
14   Brol<int> b(2);    b.t = 3;
15   b.print();
16 }
```

Affichage souhaité du code7

```
1  3 2
```

Affichage produit par le code 7

```
1  Erreur de compilation
```

Code 8

```
1  #include <iostream>
2  #include <list>
3  #include <algorithm>
4
5  class MyList
6  {
7    std::list<int> l;
8    public:
9      MyList(int i = 0) : l(std::list<int>(i)) {}
10     void add(int i) { l.push_back(i); }
11     void print()
12     {
13       for(int i : l)
14         std::cout << i << " ";
15       std::cout << std::endl;
16     }
17 };
18
19 int main()
20 {
21   std::list<MyList<int>> list(5);
22   for(auto it = list.begin(), int n = 4; it != list.end(); it++, n--)
23   {
24     MyList<int> l;
25     for(int i = n; i >= 0; i--)
26       l.add(i);
27     *it = l;
28   }
29   std::sort(list.begin(), list.end());//trie les listes par taille croissante
30   for(MyList<int> l : list)
31     l.print();
32 }
```

Nom : Prénom : Groupe : Matricule :

Affichage souhaité du code 8

```
0
0 1
0 1 2
0 1 2 3
0 1 2 3 4
```

Affichage produit par le code 8

```
Erreur de compilation
```

Code 9

```
#include <iostream>

class A
{
  int i;
  public:
    A(int i = 0) : i(i) {}
    double operator +(double d) { return i + d; }
    operator int() { return i; }
    operator double() { return i; }
};

int main()
{
  A a(2); std::cout << (a + 5) << std::endl;
}
```

Affichage souhaité du code 9

```
7
```

Affichage produit par le code 9

```
Erreur de compilation
```

Nom : Prénom : Groupe : Matricule :

Code 10

```
1  #include <iostream>
2
3  class A
4  {
5    protected:
6      int i;
7    public:
8      A(int i = 0) : i(i) {}
9      void print() { std::cout << i << std::endl; }
10 };
11
12 class B : public A
13 {
14   int j;
15   public:
16     B(int i = 0) : A(i + 2), j(i + 3) {}
17     void print() { std::cout << (i + j) << std::endl; }
18 };
19
20 int main() { A a = B(); a.print(); }
```

Affichage souhaité du code 10

```
1  5
```

Affichage produit par le code 10

```
1  2
```

Code 11

```
1  #include <iostream>
2  class A
3  {
4    public:
5      B b;
6      void print() { std::cout << "A"; }
7  };
8  class B
9  {
10   public:
11     A a;
12     void print() { std::cout << "B"; }
13 };
14
15 int main() { A a; B b; a.print(); b.print(); }
```

Affichage souhaité du code 11

```
1  AB
```

Affichage produit par le code 11

```
1  Erreur de compilation
```

Nom : Prénom : Groupe : Matricule :

Question 2. Décrivez les différentes classes d'allocations de variables en `C++`, c'est-à-dire, entre autres, /10

- en quoi elles se différencient d'un point de vue gestion de la mémoire,
- comment les mettre en œuvre (mots clés associés),
- la durée de vie, portée, création et destruction de la mémoire allouée,
- les mécanismes que certaines de ces classes rendent possibles, etc.

Question 3. Expliquez la manière dont les chaînes de caractères sont mises en œuvre en `C` pur (sans les mécanismes inhérents au `C++`), ainsi que la façon de les manipuler. /5

Question 4. Détaillez les caractéristiques qu'un problème doit posséder pour qu'une approche par programmation dynamique puisse avoir des chances de le résoudre efficacement. Pour chacune de ces caractéristiques, donnez un exemple de problème simple possédant la caractéristique et un exemple de problème simple ne la possédant pas. /15

Question 5. Vous devez stocker dans un conteneur 100 000 objets de type `A`. Pour l'application que vous développez, vous serez amenés à régulièrement extraire (obtenir sans supprimer) des éléments à des positions arbitraires de ce conteneur. Sur quel type de conteneur votre choix se porterait-il ? *Justifiez rigoureusement votre réponse.* /10

Question 6. Expliquez sommairement le principe de surcharge d'opérateur en `C++`. En particulier, détaillez /10

- les mots-clé associés,
- les limites de ce mécanisme (ce que le langage ne permet pas de faire).