

Petite révision

Quelques petits exercices pour vous rafraîchir la mémoire et vous remettre en forme. Dans les exercices pour lesquels on vous demande de programmer, cherchez à produire un résultat à la fois élégant et efficace.

Il est très probable que vous arriviez à de (bien) meilleures solutions en fin de session qu'au début en tentant de résoudre les mêmes problèmes. Je vous suggère de conserver vos résultats de début de session et de refaire l'exercice en fin de session, pour voir à quel point vous aurez cheminé cet hiver.

Un solutionnaire complet sera rendu disponible sur le site du cours une fois que vous aurez eu le temps de vous débattre un peu avec ces problèmes. Le solutionnaire n'utilisera pas les techniques qui seront enseignées durant la session, et devrait donc être à votre portée avant le début du cours.

Dans les pages qui suivent, vous trouverez parfois écrit C/ C++. Ne considérez pas cela comme s'il s'agissait d'un même langage, mais bien comme une indication du fait que la question s'applique à ces deux langages.

Q00 – Quelques petits vrai ou faux

Les questions qui suivent ont en partie trait au langage C++ mais s'appliquent aussi (en partie, du moins) au langage C.

		Vrai	Faux
Q00.0	En C++, un tableau est une structure de données.	☐	☐
Q00.1	En C++, un tableau est un objet.	☐	☐
Q00.2	Il est possible de programmer une pile sans avoir recours à de l'allocation dynamique de mémoire.	☐	☐
Q00.3	Il est possible de programmer une pile en ayant recours à de l'allocation dynamique de mémoire.	☐	☐
Q00.4	Il est possible de programmer une file sans avoir recours à de l'allocation dynamique de mémoire.	☐	☐
Q00.5	Il est possible de programmer une file en ayant recours à de l'allocation dynamique de mémoire.	☐	☐
Q00.6	Que vaut l'expression <code>0=='\0'?</code>	☐	☐
Q00.7	Qu'affichera l'extrait de code suivant?	☐	☐
	<pre>#include <iostream> int main() { using std::cout; if(3) cout << "Vrai"; else cout << "Faux"; }</pre>		
Q00.8	Qu'affichera l'extrait de code suivant?	☐	☐
	<pre>#include <iostream> int main() { using std::cout; int val = 0; if(val = 3) cout << "Vrai"; else cout << "Faux"; }</pre>		

Q01 – Typique...

Q01.0 En C++, quel est le type de chacun des littéraux suivants?

Littéral	Type
3	
"allo"	
1.0f	
-8	
"A"	
'A'	
4.0	
7L	
L"allo"	
L'A'	
3U	
3UL	
0x0a	
12.0L	
"3"s	
3s	

Q01.1 En C++, Java, C#, *etc.*, quelle est la valeur entière, exprimée en base 10, des littéraux suivants?

Littéral	Valeur
3	
0x0a	
0x000a	
0xff	
03	
8	
2.7f	
08	

Notez qu'il y a une divergence entre C# et les autres langages dans un de ces cas.

Q02 – Questions à adresser

Soit la fonction `full_cool()` qui prend en paramètre l'adresse de deux booléens, compare les booléens pointés par chacun, et retourne vrai (`true`) seulement s'ils sont de même valeur.

Q02.0 Écrivez le prototype de cette fonction.

Q02.1 Écrivez la définition de cette fonction.

Q02.2 Écrivez un petit programme qui appelle cette fonction correctement.

Q03 – Comprendre certains trucs (partie 0)...

Soit le programme C++ suivant :

```
int main() {  
    const char *texte = "Coucou";  
}
```

Combien de *bytes* sont réservés en mémoire par le compilateur pour la séquence de caractères pointée par `texte` dans cet exemple?

Q04 – Comprendre certains trucs (partie 1)...

Soit le programme C++ suivant :

```
int main() {  
    const char *texte = "Coucou";  
    texte = "Yo";  
}
```

Ce programme entraîne-t-il une fuite de mémoire? Expliquez votre réponse. Si vous estimez qu'il y a problème, comment devrions-nous le corriger?

Q05 – Comprendre certains trucs (partie 2)...

Soit le programme C++ suivant :

```
int main() {  
    int *tableau = new int[1000];  
    delete tableau;  
}
```

L'exécution de ce programme entraîne-t-elle une fuite de mémoire? Expliquez votre réponse. Si vous estimez qu'il y a problème, expliquez comment il devrait être corrigé.

Q06 – Comprendre certains trucs (partie 3)...

Soit le programme C++ suivant :

```
#include <iostream>
int main() {
    using std::cin;
    const char *t = "Wow";
    char c;
    cin >> c;
    while (c != '.') {
        t += c;
        cin >> c;
    }
}
```

Ce programme fonctionne-t-il correctement et sans danger? Expliquez votre réponse. Si vous estimez qu'il y a problème, expliquez comment il devrait être corrigé.

Q07 – Comprendre certains trucs (partie 4)...

Soit le programme C++ suivant :

```
int main() {  
    char t0[] = "Wow";  
    char t1[] = "You";  
    for (int i = 0; t1[i] != '\0'; i++)  
        t0[i] = t1[i];  
}
```

Ce programme fonctionne-t-il correctement et sans danger? Expliquez votre réponse. Si vous estimez qu'il y a problème, expliquez comment il devrait être corrigé.

Q08 – Comprendre certains trucs (partie 5)...

Soit le programme C# suivant :

```
// ...  
static void Main(string [] args)  
{  
    string s = "J'aime mon prof!";  
    // ...  
}  
// ...
```

Complétez ce programme de manière à ce qu'il contienne une `string` nommée `s2` telle que `s2` soit une copie de `s`. Expliquez ce que vous voulez dire par « une copie de ».

Q09 – Comprendre certains trucs (partie 6)...

Soit le programme C# suivant :

```
// ...
class Personne
{
    string nom;
    public string Nom
    {
        get; private set;
    }
    public Personne(string nom)
    {
        Nom = nom;
    }
}
// ...
static void Main(string [] args)
{
    string s = "Bill";
    Personne p = new Personne(s);
    s = "Bob";
    // ...
}
// ...
```

Dans la méthode `Main()`, est-ce que la modification de `s` modifiera le nom de `p`? Expliquez votre réponse.

Q10 – Quelques comparatifs

Les langages C++, Java et C# ont à la fois des similitudes importantes et des différences marquées, et il faut comprendre les unes comme les autres pour tirer profit de ces outils. Nous en examinerons plusieurs au cours de la session, mais d'ici là, jetons un coup d'œil à quelques petits trucs qui peuvent nous jouer des tours.

Q10.0

Comparez les exemples de code C++ et Java ci-dessous. Les deux font essentiellement la même chose, chacun à l'intérieur des idiomes de sa plateforme.

C++	Java
<pre>#include <string> #include <iostream> int main() { using namespace std; string texte; for (string s; cin >> s && s != ".";) texte += s; cout << texte << endl; }</pre>	<pre>import java.io.*; public class q10_0 { public static void main(String [] args) { String s = "", texte = ""; try { BufferedReader br = new BufferedReader(new InputStreamReader(System.in)); s = br.readLine(); while (!"."equals(s)) { texte = texte + s; s = br.readLine(); } } catch (IOException ex) { } System.out.println(texte); } }</pre>

Identifiez les principales différences entre ces deux programmes. Portez attention en particulier aux points suivants :

- Initialisation des variables
- Consommation de données sur un flux
- Gestion des erreurs d'accès aux flux
- Gestion de la croissance des chaînes de caractères
- Gestion de la finalisation des objets
- Comparaison de chaînes de caractères

Q10.1

Notez qu'à la question Q10.0, la version Java consomme et manipule des chaînes de caractères encodés sur 16 bits alors que la version C++ utilise des caractères encodés sur 8 bits. Apportez les modifications nécessaires à la version C++ pour qu'elle consomme elle aussi des chaînes de caractères encodés sur 16 bits.

Q10.2

Comparez les exemples de code C++ et Java ci-dessous. Les deux font essentiellement la même chose, chacun à l'intérieur des idiomes de sa plateforme.

C++	Java
<pre> #include <iostream> #include <algorithm> int main() { using namespace std; if (int n; cin >> n && n > 0) { int *p = new int[n]; for (int i = 0; i < n; ++i) cin >> p[i]; sort(p, p + n); for (int i = 0; i < n; ++i) cout << p[i] << ' '; cout << endl; delete[] p; } } </pre>	<pre> import java.io.*; import java.util.Arrays; public class q10_2 { public static void main(String [] args) { try { BufferedReader br = new BufferedReader(new InputStreamReader(System.in)); int n = Integer.parseInt(br.readLine()); if (n > 0) { int [] p = new int[n]; for (int i = 0; i < p.length; ++i) { p[i] = Integer.parseInt(br.readLine()); } Arrays.sort(p); for (int i = 0; i < p.length; ++i) { System.out.print(p[i] + " "); } System.out.println(); } } catch (IOException ioex) { } catch (NumberFormatException nfex) { } } } </pre>

Identifiez les principales différences entre ces deux programmes. Portez attention en particulier aux points suivants :

- Instanciation du tableau `p`
- Consommation d'un entier sur un flux (notez que le programme C++ tel qu'écrit comporte un petit irritant ici)
- Gestion des erreurs d'accès aux flux
- Représentation de l'algorithme de tri
- Gestion de la finalisation des objets

Q10.3

Comparez les exemples de code C++ et Java ci-dessous. Les deux font essentiellement la même chose, chacun à l'intérieur des idiomes de sa plateforme.

C++	Java
<pre>#include <iostream> #include <algorithm> int main() { using namespace std; const int N = 5; int p[N]; for (int i = 0; i < N; ++i) cin >> p[i]; sort(begin(p), end(p)); for (int i = 0; i < N; ++i) cout << p[i] << ' '; cout << endl; }</pre>	<pre>import java.io.*; import java.util.Arrays; public class q10_3 { public static void main(String [] args) { final int N = 5; try { BufferedReader br = new BufferedReader(new InputStreamReader(System.in)); int [] p = new int[N]; for (int i = 0; i < p.length; ++i) { p[i] = Integer.parseInt(br.readLine()); } Arrays.sort(p); for (int i = 0; i < p.length; ++i) { System.out.print(p[i] + " "); } System.out.println(); } catch (IOException ioex) { } catch (NumberFormatException nfex) { } } }</pre>

Identifiez les principales différences entre ces deux programmes et ceux proposés à la question Q10.2, en portant une attention particulière à l'influence d'utiliser une taille de tableau connue à l'avance dans chaque cas.

Q10.4

Plus difficile : la fonction `std::swap()`, de l'en-tête `<utility>`, est fondamentale en C++. Votre mission est d'écrire un outil équivalent en C#, et de documenter votre approche.

Q10.5

Plus difficile : la fonction `std::swap()`, de l'en-tête `<utility>`, est fondamentale en C++. Votre mission est d'écrire un outil équivalent en Java, et de documenter votre approche.