Exercices de découverte des nouveautés de c++ 2011

Objets avancés (HLIN603)

nullptr, auto, decltype, construction et initialisation des attributs d'instance

L'objectif est de tester des nouveautés de C++ 2011 au travers des TP. Certaines parties ne fonctionnent pas, essayez de comprendre pourquoi et de corriger.

Accès au compilateur le plus récemment installé : `g++-4.8`, utiliser l'option `-std=gnu++11`. La plupart des nouveautés existent cependant dans le compilateur 4.7 qui est installé par défaut (accès par `g++ -std=gnu++11`)

nullptr

`nullptr` est le pointeur `null`, unique instance du type `std::nullptr_t`. Utilisez-le dès à présent à la place de `NULL`.

Testez :

```
bool pair_int(int i){cout << "test pair/impair "; return i\%2 ==0 ;}
bool pair_int(int* i){if (!i) {cout << "pas d'entier à traiter "; return true;}
       else return (*i)\%2==0 ;}
int i ; i=2 ;
cout << pair_int(i) << endl;
int *pi = new int; *pi=3 ;
cout << pair_int(*pi) << endl;
cout << pair_int(NULL) ;
cout << pair_int(nullptr) << endl;
```

auto

C'est une indication de type qui sera remplacée tardivement par déduction issue d'une initialisation ou par spécification explicite. Quelques exemples de la documentation (indiquez les types des variables ou des expressions) :

```
auto i = 5;
const auto *v = &x, u = 6;
static auto y = 0.0;
auto ii=3;
auto g()-> decltype(ii) { return 0; }
auto h()-> decltype(ii);
```

Essayer, chercher à comprendre pourquoi ce qui suit fonctionne ou non et corriger si vous pouvez :

```
auto int j = 2;
auto h();  puis int h(){return 1 ;}
auto h();  puis auto h(){return 1 ;}
auto f(int i){if (i\%2==0) return "pair" ; else return }
auto k = 2 ; k=1.2 ;
auto l = 4.2 ; l=3 ;
auto x = 5, *y = &x;
auto a = 5, b = "cinq";
```

decltype
Pour une expression e, decltype(e) est le type de e. Quelques exemples de la documentation :

```
int i;
decltype(i) x2; // le type est int
x2 = 3 ;
```

Initialisation et construction Trois nouveautés vont nous intéresser :

- Les attributs d'instance peuvent être initialisés lors de leur déclaration.
- Un constructeur peut déléguer à un autre constructeur de la même classe.
- Un objet peut être initialisé lors de sa création par une liste d'initialisation (que l'on reverra dans le cas des collections).

Repérez ces éléments dans le programme suivant :

```
#include<iostream>
#include<string>
using namespace std;

class Date {
private:
int jour = 1, mois = 1, annee = 2000;
public:
Date();
Date(int j, int m, int a);
};

Date::Date():Date(1,1,1970){}
Date::Date(int j, int m, int a){
this->jour=j; this->mois=m; this->annee=a;
}

main()
{
Date d1(15,1,2014);
Date d2 = {16,1,2014};
}
```

Sources
working draft ISO/IEC JTC1 SC22 WG21 N3690 2013-05-15 Programming Languages - C ++
http ://isocpp.org/files/papers/N3690.pdf

La présentation de Jean-Paul Rigault, Professeur à l'Université de Nice Sophia Antipolis
http ://www.polytech.unice.fr/~jpr/c++2011/

Le site de Bjarne Stroustrup, Créateur de C++
http ://www.stroustrup.com/C++11FAQ.html

Le site du compilateur Gnu indiquant quelles caractéristiques de C++ sont prises en charge par les différentes versions
http ://gcc.gnu.org/projects/cxx0x.html