Initiation aux concepts objets en C++

Du C au C++ - Découverte du paradigme objet

Sommaire

Sommaire (détail)

- II. Principales différences entre le C et le C++
  A. Les flux d'entrée et sortie
  B. Les types scalaires
  C. La déclaration des variables
  D. La gestion de la mémoire
  E. Les paramètres par défaut
  F. La surcharge de fonctions
  G. Les fonctions « inline »
  H. Les chaines de caractères

Objectifs

- Appréhender les premières lignes de C++
- Comprendre les différences entre le C et C++

Du C au C++

- Les flux d'entrée et de sortie

En langage C	En langage C++
stdout	cout
stdin	cin
stderr	cerr

- Les fonctions d'entrée/sortie

En langage C	En langage C++
printf	<<
scanf	>>

Du C au C++

- Exemple C-1 : Saisie d'un entier et d'une chaine de caractères avec affichage

```
4   void input()
5   {
6       int age = 0;
7       char name[101];
8       printf("Enter your age : \n");
9       scanf("%d", &age);
10      printf("Enter your name (100 char max) : \n");
11      scanf("%s", name);
12      printf("\nYou are %d years old and your name is %s\n", age, name);
13  }
```

Du C au C++

- Exemple CPP-1 : Saisie d'un entier et d'une chaine de caractères avec affichage

```
4   void input()
5   {
6       int age = 0;
7       char nom[101];
8       std::cout << "Enter your age : " << std::endl;
9       std::cin >> age;
10      std::cout << "Enter your name (100 char max) : " << std::endl;
11      std::cin >> nom;
12      std::cout << std::endl << "You are " << age
13                << " years old and your name is " << nom << std::endl;
14  }
```

Du C au C++

- Les types scalaires
  - Entier : int
  - Caractère : char
  - Réel : float, double
  - Pointeur : *
  - **Référence : & (NOUVEAUTE en C++)**
  - **Booléen : bool (NOUVEAUTE en C++)**

Du C au C++

- Exemple C-2 : Utilisation du booléen

```
 4  void boolean()
 5  {
 6      unsigned char isAdult = 0; // by default : no
 7      int age = 0;
 8      printf("Enter your age : \n");
 9      scanf("%d", &age);
10      if (age >= 18)
11      {
12          isAdult = 1; // you are an adult
13          printf("Adult = true\n");
14      }
15      else
16      {
17          printf("Adult = false\n");
18      }
19  }
```

Du C au C++

- Exemple CPP-2 : Utilisation du booléen

```
 4   void boolean()
 5   {
 6       bool isAdult = false; // by default : no
 7       int age = 0;
 8       std::cout << "Enter your age : " << std::endl;
 9       std::cin >> age;
10       if (age >= 18)
11       {
12           isAdult = true; // you are an adult
13           std::cout << "Adult = true" << std::endl;
14       }
15       else
16       {
17           std::cout << "Adult = false" << std::endl;
18       }
19   }
```

Du C au C++

- Déclaration des variables
  - Au milieu d'une fonction (possible aussi avec une version récente du C)
  - Dans une boucle ☺

Du C au C++

- Exemple C-3 : Déclaration de variable dans boucle => **IMPOSSIBLE en C**

```
 4   void loop()
 5   {
 6       int i=0;
 7       for (i=0; i<10; i++)
 8       {
 9           printf("Loop #%d\n", i);
10       }
11   }
```

Du C au C++

- Exemple CPP-3 : Déclaration de variable dans boucle

```
3   void loop()
4   {
5       for (int i=0; i<10; i++)
6       {
7           std::cout << "Loop #" << i << std::endl;
8       }
9   }
```

Du C au C++

- Allocation dynamique de mémoire

En langage C	En langage C++
malloc	new
free	delete

- Pour les tableaux : new[] et delete[]

- **Attention : new et delete sont des opérateurs !**

Du C au C++

- Exemple C-4 : Allocation mémoire

```
 4   void memory()
 5   {
 6       int i = 0;
 7       int* tab = (int*)malloc(5*sizeof(int));
 8       for (i=0; i<5; i++)
 9       {
10           tab[i] = rand()%20;
11       }
12       for (i=0; i<5; i++)
13       {
14           printf("tab[%d] = %2d\n", i, tab[i]);
15       }
16       free(tab); // Do not forget to free memory !!
17   }
```

Du C au C++

- Exemple CPP-4 : Allocation mémoire

```
 4  void memory()
 5  {
 6      int* dummy = new int;
 7      int* tab = new int[5];
 8      for (int i=0; i<5; i++)
 9      {
10          tab[i] = rand()%20;
11      }
12      for (int i=0; i<5; i++)
13      {
14          std::cout << "tab[" << i << "] = " << tab[i] << std::endl;
15      }
16      delete dummy; // do not use [] because dummy is not an array
17      delete[] tab; // Do not forget to free memory !!
18  }
```

Du C au C++

- Déclaration des structures/énumérations
  - Typedef automatique
- Déclaration des unions
  - Typedef automatique

Du C au C++

- Exemple C-5 : Déclaration d'une structure

```
4   typedef struct coordonnees coordonnees;
5   struct coordonnees
6   {
7       int x,y;
8   };
9
10  void useCoord()
11  {
12      coordonnees* c = (coordonnees*)malloc(sizeof(coordonnees));
13      c->x = 0;
14      c->y = 0;
15      free(c);
16  }
```

Du C au C++

- Exemple CPP-5 : Déclaration d'une structure

```
4   struct coordonnees
5   {
6       int x,y;
7   };
8
9   void useCoord()
10  {
11      coordonnees* c = new coordonnees;
12      c->x = 0;
13      c->y = 0;
14      delete c;
15  }
```

Du C au C++

- Pour les fonctions :
  - Paramètres par défaut acceptés
  - Surcharges de fonction acceptées
  - Fonctions « inline »

Du C au C++

- Exemple C-6 : Fonction avec valeur par défaut

```
4   int add(int a, int b, int c = 0)
5   {
6       return (a + b + c);
7   }
```

➢ **IMPOSSIBLE EN C !! (ne compile pas)**

Du C au C++

- Exemple CPP-6 : Fonction avec valeur par défaut

```
4   int add(int a, int b, int c = 0)
5   {
6       return (a + b + c);
7   }
```

  - Ici le 3ème paramètre « c » est facultatif

- Appel du sous programme « add »

```
std::cout << add(1,2,3) << std::endl; // here we have 3 params
std::cout << add(1,2) << std::endl; // here, we have 2 params
```

Du C au C++

- Exemple C-7 : Surcharge de fonction

```
4   // Multiplication for integer
5   int multiply(int nb1, int nb2)
6   {
7       return (nb1 * nb2);
8   }
9
10  // Multiplication for double
11  double multiply(double nb1, double nb2)
12  {
13      return (nb1 * nb2);
14  }
```

➢ **IMPOSSIBLE EN C !! (ne compile pas)**

Du C au C++

- Exemple CPP-7 : Surcharge de fonction

```
4   // Multiplication for integer
5   int multiply(int nb1, int nb2)
6   {
7       return (nb1 * nb2);
8   }
9
10  // Multiplication for double
11  double multiply(double nb1, double nb2)
12  {
13      return (nb1 * nb2);
14  }
```

Du C au C++

- Exemple CPP-7 : Surcharge de fonction (suite)

```
// call multiply for integer
std::cout << multiply(3,7) << std::endl;

// call multiply for double
std::cout << multiply(3.0,7.0) << std::endl;
```

Du C au C++

- Exemple C-8 : Inline et fonctions

```
4   inline int max(int a, int b)
5   {
6       return (a > b ? a : b);
7   }
```

➢ **Possible en C depuis la norme « C99 »**

Du C au C++

- Exemple CPP-8 : Inline et fonctions

```
4   inline int max(int a, int b)
5   {
6       return (a > b ? a : b);
7   }
```

- **Avantage : rapidité d'exécution**
- **Désavantage : code plus gros en taille**

Du C au C++

- Les chaines de caractères

En langage C	En langage C++
strlen	String.size()
strcpy / strncpy	Opérateur =
strcat / strncat	Opérateur +=
strcmp / strncmp	Opérateurs ==, >=, <=, >, <

- Enfin un objet « String »
  - Opérations élémentaires très simplifiées en C++ !!

Du C au C++

- Exemple C-9 : Chaines de caractères

```
void useString()
{
    char buffer[100], bufferCopy[100];
    strcpy(buffer, "Hello ");
    strcat(buffer, "World"); // concatenate two strings
    printf("%s [Size=%d]\n", buffer, strlen(buffer));
    strcpy(bufferCopy, buffer); // copy a string
    printf("%s [Size=%d]\n", bufferCopy, strlen(bufferCopy));
}
```

- **Attention aux débordements de tableau…**

Du C au C++

- Exemple CPP-9 : Chaines des caractères

```
void useString()
{
    std::string str, strCopy;
    str = "Hello ";
    str += "World"; // concatenate two strings
    std::cout << str << " [Size=" << str.size() << "]" << std::endl;
    strCopy = str; // copy a string !
    std::cout << strCopy << " [Size=" << strCopy.size() << "]" << std::endl;
}
```

- Avantage énorme !
  - **Plus besoin de vérifier les débordements !!**

Prochaine séance

- Découvertes des classes et objets