2

Lecture/écriture des variables

Dans la section précédente, nous avons appris comment :

1. déclarer une variable :

```
int a;
```

a ?

2. initialiser une variable :

```
int b = 5;
```

b 5

3. affecter une variable :

```
a = 2 * b + 3;
```

a 13

Dans cette partie, nous allons nous intéresser à comment :

- Afficher (écrire) la valeur d'une variable à l'écran.
- Lire une variable à partir du clavier.

1. Affichage (écriture) des variables

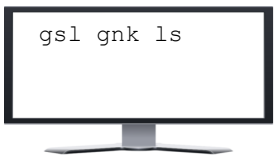
En C++, pour afficher un message à l'écran, on écrit `cout <<` suivi par le message.

Exemple 1 : Le programme suivant affiche `"Hello"` à l'écran.

```
#include<iostream>
using namespace std;
int main(){
	cout << "Hello!";

	return 0;
}
```

`Cout (console output)` : permet d'afficher à l'écran ce qui suit le symbole "<<". En effet, ce qui est entre guillemets (") est affiché littéralement, même s'il n'a aucun sens.

```
#include<iostream>
using namespace std;
int main(){
	cout  << "gsl gnk ls";

	return 0;
}
```



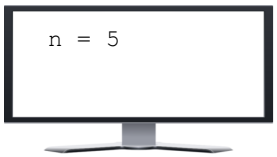
Exemple 2

```
int n = 5;

cout  << "n";

cout << n;
```

On remarque bien que `cout << "n";` et `cout << n;` ne font pas la même chose. La première affiche la lettre `"n"` alors que la deuxième affiche la valeur contenue dans la variable `n`.

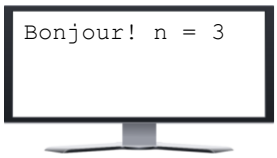
Exemple 3 : On peut combiner l'affichage d'un message avec celui d'une variable en les séparant par le symbole <<:

```
int n = 5;
cout  << "n = " << n;
```



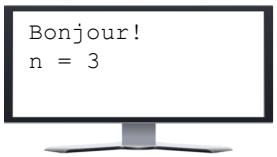
Exemple 4 : On peut afficher à plusieurs endroits dans le programme:

```
cout << "Bonjour! ";
int n = 3;
cout  << "n = " << n;
```



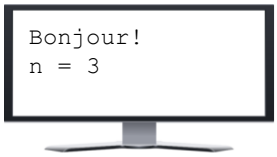
Pour revenir à la ligne, on peut utiliser `endl` qui veut dire `end of line`, comme suit:

```
cout << "Bonjour!" << endl;
int n = 3;
cout  << "n = " << n;
```



Comme on peut utiliser `"\n"` pour revenir à la ligne (au lieu de `endl`):

```
cout << "Bonjour! \n";
int n = 3;
cout  << "n = " << n;
```

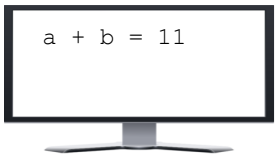


Exemple 5 : On peut utiliser `cout` pour afficher une valeur:

```
cout  << 2;
```

Comme on peut utiliser `cout` pour afficher la valeur d'une expression :

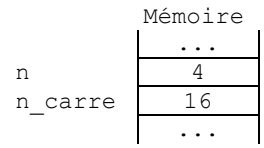
```
int a = 5;
int b = 6;
cout  << " a + b = " << a + b;
```

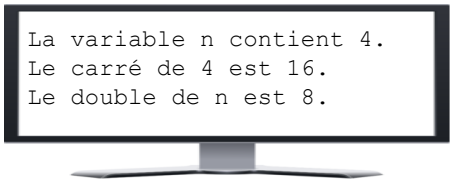


Exemple 6 : Soit le programme suivant:

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

int main (){
       int n = 4;
       int n_carre;
       n_carre = n * n;

       cout << "La variable n contient " << n << "." << endl;
       cout << "Le carré de " << n << " est " << n_carre << "." << endl;
       cout << "Le double de n est " << 2 * n << "." << endl;

       return 0;
}
```

Déroulement du programme





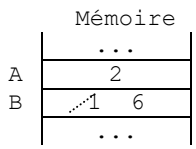
Remarque : On peut écrire `cout` et `endl` simplement car le début du programme contenait la ligne:
`using namespace std;`

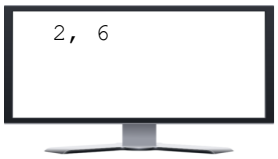
En cas d'absence de cette ligne, il faut écrire `std::cout` et `std::endl`.

```
#include<iostream>

int main (){
       int n = 4;
       int n_carre;
       n_carre = n * n;

       std::cout << "La variable n contient " << n << "." << std::endl;
       std::cout << "Le double de n est " << 2 * n << "." << std::endl;

       return 0;
}
```

Exercice 1 : Qu'affiche le programme C++ suivant :

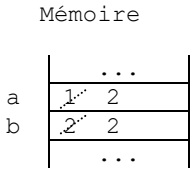
```
#include<iostream>
using namespace std;
int main (){
       int a = 2;
       int b = 1;
       b = a * (b + 2);

       cout << a << ", " << b;

       return 0;
}
```





Exercice 2 : Qu'affiche le programme suivant :

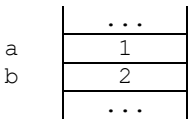
```
#include<iostream>
using namespace std;
int main (){
       int a = 1;
       int b = 2;
       a = b;
      b = a;

       cout << a << ", " << b;
       return 0;
}
```



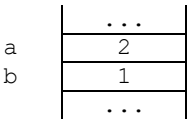
Exercice 3: Ecrire une programme qui permet d'échanger les valeurs de deux variables**.**

Solution : En effet, dans le programme de l'exercice 2, on voulait échanger les valeurs des variables `a` et `b`:

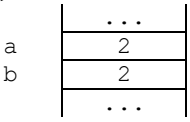
```
int a = 1;
int b = 2;
```



Cependant, on n'a pas obtenu le résultat voulu. On voulait obtenir :



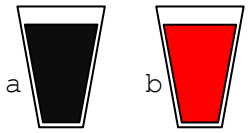
Mais, on a obtenu :



On essayé d'échanger leurs valeurs comme suit:

```
a = b;
b = a;
```

Pourquoi ça n'a pas marché? Pour comprendre pourquoi ça n'a pas marché, on peut assimiler l'échange des valeurs de deux variables à deux verres, l'un par exemple contient du café et l'autre contient de la grenadine :



Bien évidemment, pour échanger le contenu de ces deux verres, on a besoin d'un troisième verre vide:

Le même principe s'applique aux variables. Alors, pour échanger les valeurs de deux variables, on doit utiliser une variable intermédiaire. On peut procéder, par exemple, comme suit: on stocke (copie) la valeur de `a` dans la variable intermédiaire `c`. Ensuite, on copie la valeur de `b` dans `a` (ce qui écrase la valeur de `a`). Enfin, on copie la valeur stockée dans `c` dans la variable `b`.

```
#include<iostream>
using namespace std;
int main (){
       int a = 1;
       int b = 2;

      int c = a;
       a = b;
      b = c;
       cout << a << ", " << b;
       return 0;
}
```

Déroulement du programme

```
int a = 1;
int b = 2;
int c = a;
```

	...
a	1
b	2
c	1
	...

```
a = b;
```

	...
a	2
b	2
c	1
	...

```
b = c;
```

	...
a	2
b	1
c	1
	...

2. Lecture des variables

Soit le programme suivant:

```
#include<iostream>
using namespace std;
int main (){
    int n = 4;
    int n_carre = n * n;
    cout << "Le carré de " << n << " est " << n_carre << "." << endl;
    return 0;
}
```

Que fait ce programme? En fait, ce programme ne calcule que le carré de 4. En d'autres, ce programme n'a pas une grande utilité. Pour le rendre plus utile, on peut demander à l'utilisateur de donner la valeur pour laquelle il veut calculer le carré.

Pour ce faire, on doit faire le changement suivant :

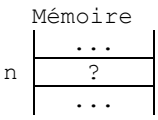
```
#include<iostream>
using namespace std;
int main (){
   int n;
   cout << "Entrez un nombre: ";
   cin >> n;
   int n_carre = n * n;
    cout << "Le carré de " << n << " est " << n_carre << "." << endl;
    return 0;
}
```

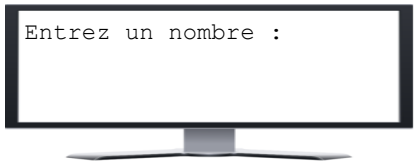
`Cin (console input)` : permet de lire une valeur à partir du clavier.

`cin >> n;` permet de lire une valeur à partir du clavier et la mettre dans la variable `n`. Notez bien le sens de la flèche (>>) qui signifie mettre une valeur dans la variable.

Exécutons le programme pas à pas :

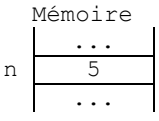
```
int n;
cout << "Entrez un nombre: ";
cin >> n;
```

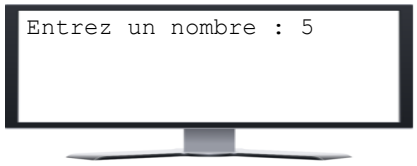




L'utilisateur doit saisir un nombre et appuyer sur la touche Entrée du clavier, sinon le programme ne ferra rien (il attend la valeur de `n`).

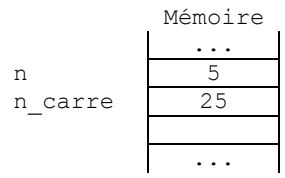
Supposons que l'utilisateur a choisi `5`.

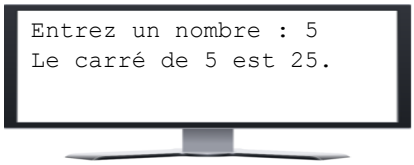




Maintenant, l'ordinateur pourra continuer l'exécution du reste du code.

```
int n_carre;
n_carre = n * n;
cout << "Le carré de " << n << " est " << n_carre << "." << endl;
```





Remarque 1 : Uniquement les variables peuvent figurer à droite du symbole >>. En d'autres termes, on ne peut pas écrire:

```
cin >> 5;
cin >> 2 * n;
cin >> "Entrez un nombre : ";
cin >> "Entrez un nombre : " >> n;
```

Si on veut afficher un message avant de lire une valeur, on doit procéder comme suit:

```
cout << "Entrez un nombre: ";
cin >> n;
```

Remarque 2 : On peut lire plusieurs valeurs à la suite:

```
int a, b, c;
cin >> a >> b >> c;
```

Mais, on préfère l'écriture suivante:

```
int a, b, c;
cin >> a;
cin >> b;
cin >> c;
```