INFO-H-100
Séance d'exercices 1

Les instructions élémentaires

INFO-H-100

- http://cs.ulb.ac.be/public/teaching/infoh100

- Professeur :
  - Thierry Massart
- Assistants :
  - Julien Roland
  - Frédéric Servais
  - Alain Silovy
  - Boris Verhaegen

TPs

- Rappels théoriques : slides sur le site
- Exercices : corrigés sur le site
- Trucs & astuces vus au TP et aux projets sont matière d'examen
- TPs sur papier
- Séances 2, 4 et 6 sur machine
  - Activer votre compte ULB
  - Aller à la salle informatique pour ces séances

Elèves assistants

- Présents les midis à partir de la semaine prochaine dans la salle Socrate.
- Pour toutes questions : C++, compilation, projets, exercices.
- Ces jeudi et vendredi midi à la salle Socrate : Introduction aux salles.
  - Linux
  - Compilation
  - Terminal

Projets & examen

- 2 projets :
  - 1/5 des points du cours de programmation
  - Projet 1 : 14 au 28 mars
  - Projet 2 : 4 avril au 9 mai
- Examen :
  - Examen d'exercices sur papier
  - Niveau des exercices les plus difficiles vus aux TPs
  - Séances 10 et 12 consacrées aux anciens examens

INFO-H-100
Séance d'exercices 1

Les instructions élémentaires

Entités

- **Variable** : sa valeur peut changer.
  - Type
  - Nom
  - Espace mémoire
- **Littéral** : sa valeur est fixe.
  - Type
  - Valeur
  - Exemples : `15, true, ‘b’, ‘ ‘, “bonjour”, 3.14`

Variables

- **Type** : défini le type de valeur contenu à l'espace mémoire spécifié.
  - Entier (`int`)
  - Réel (`double`)
  - Booléen (`bool`)
  - Caractère (`char`)
- **Espace mémoire** : endroit dans la mémoire centrale où est stoquée la valeur de la variable.
- **Nom** : permet de manipuler (lire/écrire) la valeur écrite à l'espace mémoire spécifié.
  - Par exemple : `Tva, compteur, résultat`, …

Déclarations de variables

- Type et nom
- Actions :
  - Réserve un **espace mémoire** de taille spécifiée par le type
  - Lie le nom à cet espace mémoire
- Exemples :
  - `int compteur; bool b; double Tva; char c;`
- La déclaration d'une variable se fait toujours avant son utilisation
- Après sa déclaration, la valeur d'une variable est indéterminée.
- Choisir un nom signifiant

Opérateurs

- Arithmétiques :
  - `+, -, *, /, %`
  - Pas d'opérateur exposant
- Logiques :
  - `not, and, or (ou !, &&, ||)`
- Relationnels :
  - `==, !=, <, <=, >, >=`
- Remarque :
  - `a < b < c` n'est pas correct
  - `a < b and b < c` est correct

Expression

- Une expression dénote une valeur. Elle est formée d'**opérateurs** et d'**opérands**.
- Exemples :
  - `a`
  - `3`
  - `a*3`
  - `b`
  - `(3+a) == b`
- On peut composer de nouvelles expressions à partir d'expressions et d'opérateurs.

Valeur d'une expression

- La valeur est calculée à partir des valeurs des variables **au moment de l'évaluation**.
- Exemple : `(3+a) == b`
- Règles d'évaluation :
  - Priorités des opérateurs
  - Parenthèses
  - Ordre d'évaluation

Exemples d'évaluations

- `(3+a) == b`
- `((21+36)*37) + (47+(38+58))`
- `true and x<14`

a	b	a and b	a or b
true	true	true	true
true	false	false	true
false	true	false	true
false	false	false	false

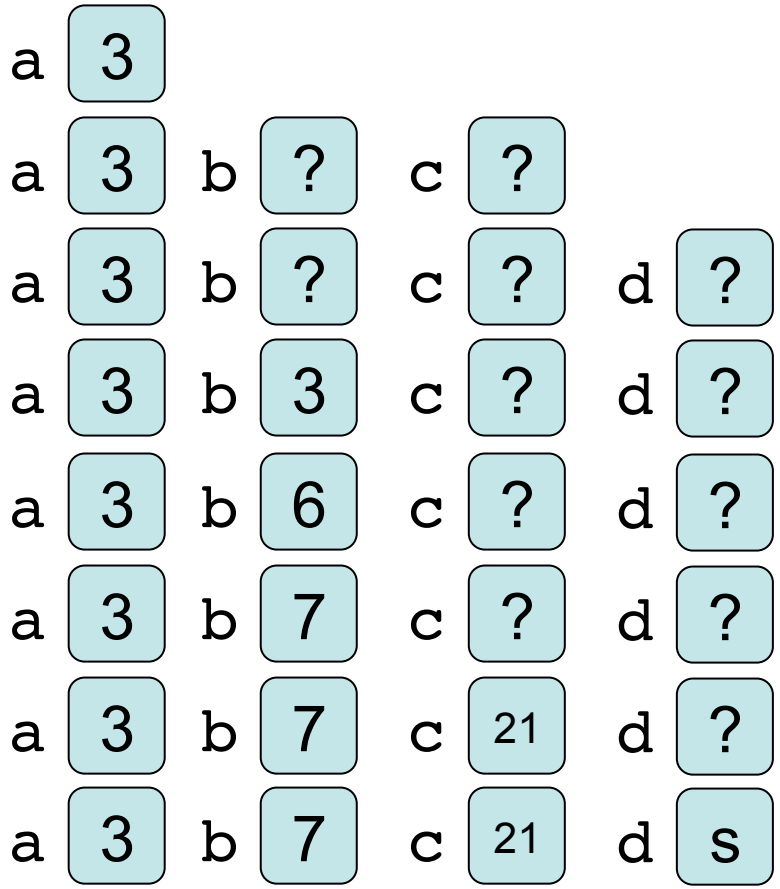
Assignation

```
<variable> = <expression> ;
```

- “=“ se dit “reçoit”
- Actions :
  1. évaluer l’expression
  2. écrire le résultat de l’évaluation dans l’espace mémoire de la variable.
- Le type de l’expression doit être compatible avec le type de la variable
- Attention :
  - aux variables déclarées mais non-initialisées
  - `a=b` n’est pas `a==b`
- Déclarer et initialiser une variable : `int a = 3;`

Exemples d'assignations

```
int a = 3;
int b, c;
char d;
b = a;
b = a+3;
b = b+1;
c = a*b;
d = 's';
```



Lecture et écriture

- Entrée (*input*) clavier avec `cin` :
  - `cin >> x ;`
  - `cin >> a >> b ;`
  - on lit une **variable**

- Sortie (*output*) à l'écran avec `cout` :
  - `cout << a ;`
  - `cout << "La valeur de a est " << a ;`
  - `cout << a + 5 << endl ;`
  - `cout << 'c' << endl;`
  - on affiche la **valeur d'une expression**

Structure d'un programme

- Instruction simple : expression + ';'
- Exécution séquentielle des instructions
- Structure générale

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


int main()
{


   … instructions …


}
```

Exercices

- 1
- 3
- 4
- 5
- 6
- 18
- 14 (demo)
- 16 (demo)