Les entrées/sorties

Clavier / Terminal : `cin` / `cout` et `cerr`

Fichier de définitions : `#include <iostream>`

Utilisation :

- écriture : `cout << expr1 << expr2 << ... ;`
- lecture : `cin >> var1 >> var2 >> ... ;`

Saut à la ligne : `endl`

Lecture d'une ligne entière : `getline(cin, string);`

.....

Formatage :

Manipulateurs	
`#include <iomanip>`	
`cout << manip << expr << ...`	
`dec`, `oct`, `hex`	changement de base
`setprecision(int)`	nombre de chiffres à afficher
`setw(int)`	largeur de colonne
	nombre de caractères à lire
`setfill(char)`	caractère utilisé dans l'alignement
`cin >> ws`	saute les blancs

Options	
`setf(ios::option)`	
`unsetf(ios::option)`	
`ios::left`	alignement à gauche
`ios::showbase`	afficher la base
`ios::showpoint`	afficher toujours la virgule
`ios::fixed`	notation fixe
`ios::scientific`	notation scientifique

Jean-Cédric Chappelier
& Jamila Sam
EPFL

Les entrées/sorties (2)

Fichiers : `#include <fstream>`

Flot d'**entrée** (similaire à `cin`) : `ifstream`
Flot de **sortie** (similaire à `cout`) : `ofstream`

Création : *type_flot nom_de_flot*;

Lien (ouverture) : *flot*.open("*fichier*");


ouverture en binaire :

pour lecture : ifstream *flot*("*fichier*", ios::in|ios::binary);
pour écriture : ofstream *flot*("*fichier*", ios::out|ios::binary);

Utilisation : comme `cin` et `cout` :

flot << *expression* << ... ;
flot >> *variable_lue* >> ...;

Test d'échec de ouverture/lecture/écriture sur le flot : *flot*.fail()

Fermeture du fichier : *flot*.close()

Test de fin de fichier : *flot*.eof()

Jean-Cédric Chappelier
& Jamila Sam
EPFL