POLYTECH SORBONNE

Automne 2019

Flowchart/AsciiArt

TP Solo

Nom : **Prénom :**

Contraintes

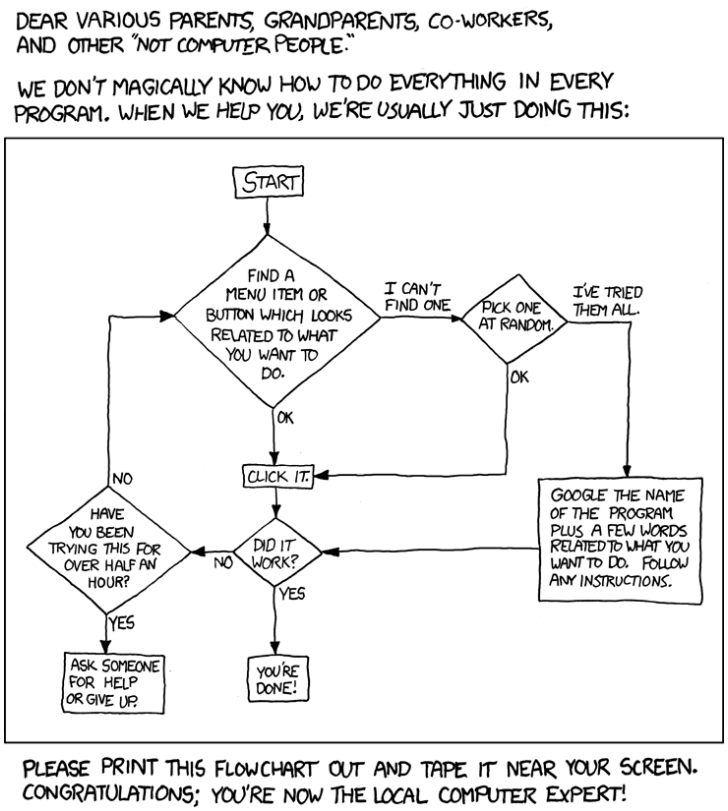
- Indentez vos fichier.
- La correction tiendra compte de la brièveté des méthodes que vous écrivez (évitez les fonctions de plus de 25 lignes) ; n'hésitez pas à découper une méthodes en plusieurs sous-méthodes (privées) plus courtes.
- Votre code ne doit pas donner d'erreurs avec Valgrind (ni fuite mémoire, ni autre erreurs).
- Vous ne devez pas utiliser de fonction C quand un équivalent C++ existe.
- Les noms de classe commencent par une majuscule.
- Les noms de méthodes et d'attributs commencent par une minuscule.
- Vous devez fournir un Makefile qui compile vos fichiers source et contient une règle clean ainsi qu'un programme de test.
- Le **code source et les diagrammes** doivent être *pusher* sur le git du TP

Préparation du TP

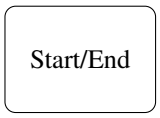
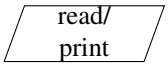
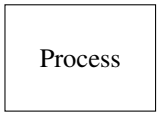
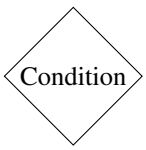
- Cloner votre répertoire sur votre compte
  `git clone https://git` l'adresse qui vous a été attribué.
- Pendant le TP n'oublier pas de comité régulièrement
- Le répertoire contient un fichier un ensemble de fichiers qui vous serviront.
- N'oublier pas de pusher l'ensemble

Le dépôt git ne doit pas contenir d'exécutable, ni de fichier objet, ni de fichier temporaire (*.~)

Concept



Les diagramme de flow ou “flowchart” sont une représentation graphique d’un algorithme. Ils peuvent être très utile pour concevoir des algorithmes ou expliquer ses algorithmes. Différents symboles normalisés sont utilisés pour décrire des programmes. Un sous ensemble de ces symboles utilisés pour ce TP sont représentés dans le tableau suivant.

Symbole	Fonction	Description
	Ligne de flux	Utiliser pour indiquer le flux qui connecte les symboles
	Terminal (Start/End)	Utiliser pour indiquer le début et la fin du flowchart
	Input/Output	Utiliser pour les opérations d’entrées/sorties (lecture/ecriture)
	Processing	Utiliser pour les opérations arithmétiques, affectations ...
	Decision	Utiliser pour les alternatives (vraies ou fausses).

Nous avons choisit de modéliser un flowchart comme un ensemble de forme et un ensemble de connections entre ces formes. L’objectif de ce TP est de réaliser la modélisation d’un flowchart et de ce sous-ensemble de symboles.

Dans ce TP, nous allons nous concentrer sur la modélisation et l’implémentation des symboles.

1 Modélisation UML (30 minutes max.)

**** L'implémentation est l'objet de la question suivante ****

Proposez une modélisation UML qui puisse répondre à ce problème. Vous devez respecter l'esprit de la modélisation objet et, notamment, utiliser l'héritage à bon escient. De plus **lisez les fichiers fournis pour être compatible !**

Votre proposition UML doit permettre de réaliser le test suivant.

```
void test1()
{
  Start * s1 = new Start;
  End * s2 = new End;
  Decision * s6 = new Decision("d1","a<b");
  Decision * s7 = new Decision(*s6);
  s7->setText("b>=a");
  std::cout << "\""<< s6->label() << "\" (id:" << s6->id() << ") Nombre de connexions ";
  std::cout << s6->nb_connect_max() << std::endl;
  std::cout << "\""<< s7->label() << "\" (id:" << s7->id() << ") Nombre de connexions ";
  std::cout << s7->nb_connect_max() << std::endl;


  std::cout << s1->toString() << std::endl;
  std::cout << s2->toString() << std::endl;
  std::cout << s6->toString() << std::endl;
  std::cout << s7->toString() << std::endl;
}
```

Votre diagramme UML

2 Implémentation

Le code que nous vous fournissons contient beaucoup d'erreurs. Vous pouvez constatez que bien qu'un Makefile soit fournit, le programme ne compile pas.
Le résultat attendu est le suivant:

```
"d1" (id:2) Nombre de connexions 2
"d1" (id:3) Nombre de connexions 2
 -----
/     \
|Start|
\_____/

 ---
/   \
|End|
\___/

  ^
 / \
/a<b\
\   /
 \ /
  V

  /\
 /  \
/b>=a\
\    /
 \  /
  \/
```

1. Corriger le programme pour avoir un code qui compile sans erreur. Pour chacune des erreurs corrigées veuillez écrire ci-après quelle est la cause de l'erreur et quelle correction avez-vous apporté.
2. Compléter le code pour obtenir le même affichage (*attention aux nombres de connexions*) et expliquez vos changements.

3 Création d'une classe *InputOutput*

Vous devez maintenant écrire la classe *InputOutput* compatible avec le `test2()` et qui doit produire le resultat suivant :

```
"Declaration 1" (id:2) Nombre de connexions 1
"d1" (id:5) Nombre de connexions 2
"d1" (id:6) Nombre de connexions 2
 -----
/     \
|Start|
\_____/

 ---
/   \
|End|
\___/

 ___________
| declare a |
| declare b |
| declare c |
 -----------

   _________
  / read a /
 / read b /
 ---------

    ___________
   /  print a /
  /  print b /
 /  print c /
 -----------

  ^
 / \
/a<b\
\   /
 \ /
  V

  /\
 /  \
/b>=a\
\    /
 \  /
  \/
```