Projet Application C++ Calculatrice HP

Simulation d'une calculatrice HP à notation postfixée

Ce projet long est un exercice noté en binôme. À présenter lors de la dernière session de cours, le 29 Mai.

Objectif

Développer une application C++ pour simuler les fonctionnalités d'une calculatrice HP à notation postfixée. Il s'agira d'effectuer une application avec interface graphique.
L'application doit être la plus simple d'utilisation possible, avec un certain nombre de fonctionnalités pratiques pour une calculatrice et une exactitude des résultats.
Il est très important de collaborer sur le projet avec BitBucket et de faire des commits réguliers, notamment à chaque étape (paragraphe) du projet. Les commits équilibrés des deux membres du binômes seront vérifiés. Le framework Catch sera utilisé pour tester unitairement les classes et fonctions (non graphiques) du programme.
La calculatrice pourrait avoir l'apparence donnée par la Fig.1. (ou mieux selon vos ambitions à utiliser une autre bibliothèque graphique)

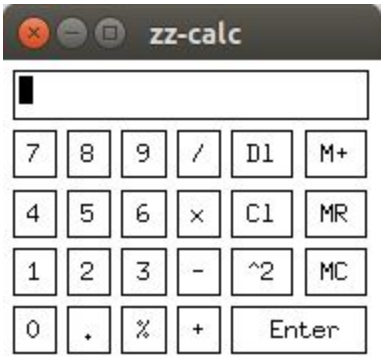

Fig.1: Interface graphique de la calculatrice

1. Commencer avec la bibliothèque `libsx`

L'exemple de la Fig.1 est effectuée à l'aide de la bibliothèque **`libsx`**.
Elle devra être installée sur votre environnement linux. Si ce n'est pas déjà le cas, obtenez la en effectuant la commande suivante dans le Terminal:

```
sudo apt-get update && sudo apt-get install libsx-dev libsx0
```

Une fois l'installation terminée avec succès, la documentation se trouve ici :

```
/usr/share/doc/libsx-dev/html/libsx.html
```

La bibliothèque **`libsx`** est très simple à utiliser. Elle offre des composants graphiques préfabriqués (boutons, zone de texte, menu déroulant, etc ...) appelés *Widgets*. Il suffit de les assembler pour composer l'interface graphique.
L'assemblage des composants graphiques se fait en deux étapes :

- disposition des composants graphiques
- connexion des composants graphiques avec les actions

1.a. Partie graphique

Pour l'application de la Fig.1, nous avons utilisé les composants entrées de texte et boutons. Une entrée de texte est une simple zone graphique qui permet l'édition d'un message de type chaîne de caractères. Elle va nous servir pour visualiser et éditer la valeur courante traitée par la calculatrice.

La fonction `MakeStringEntry` crée une entrée. Elle possède quatre arguments. Son prototype est le suivant :

```
Widget MakeStringEntry(char *txt, int size, StringCB func, void *data);
```

Le deuxième argument est la taille en pixels de l'entrée.
Le troisième argument est la callback (fonction action) à exécuter. On peut mettre `NULL` s'il y en a pas.
Le quatrième argument sera utilisé pour passer des données à la fonction. On peut mettre `NULL` s'il y en a pas.

La fonction `MakeButton` crée un bouton. Son prototype est le suivant :

```
Widget MakeButton(char *txt, ButtonCB func, void *data);
```

Le premier argument correspond au texte à afficher sur le bouton.
Le deuxième argument est la callback (fonction action) à exécuter. On peut mettre `NULL` s'il y en a pas.

Créer l'interface graphique de la calculatrice.
Note: utiliser un Makefile pour la compilation du programme.

1.b Partie calcul

Pour évaluer une expression postfixée, le programme a besoin d'une pile. L'exemple suivant nous explique comment ça marche. L'expression est parcourue de gauche à droite. Chaque opérande est empilé et un opérateur trouve ses deux opérandes en sommet et sous-sommet de pile, qu'il remplace par le résultat de l'opération.

Exemple:

notation préfixée	notation postfixée	résultat
1 + 3	1 3 +	4
2 + 3 x 4	2 3 4 x +	14
2 x 3 - 4 x 5	2 3 x 4 5 x -	-14

Créer les classes qui permettent d'effectuer ces calculs et les intégrer à la partie graphique.
Pour empiler les opérandes, on définira une classe `Pile` contenant un attribut de type stack (conteneur C++)
Il faudra utiliser le framework Catch pour tester unitairement chaque classe.

2. Pour aller plus loin

Utiliser une bibliothèque graphique C++ de votre choix pour effectuer la partie graphique et la connecter à la partie calcul. Ajouter plus de fonctionnalités à votre calculatrice afin qu'elle soit le plus utile possible.
Note: Faire des commits réguliers (et notamment quand une fonctionnalité est opérationnelle.)