C-- : Guide utilisateur

H4222

C-- est un compilateur d'un sous-ensemble du langage C réalisé avec Antlr4 et C++.

Instructions

1. Compiler l' environnement

Se placer dans le dossier "compiler" et exécuter un des 3 scripts ci-dessous :

- `Makefile` à l'aide de la commande 'make' pour compiler lr parser et ajouter les librairies et les dépendances sur les machines des salles 208 and 219 de IF.
- `compile_ubuntu.sh` pour compiler sur une machine a ubuntu where le antlr runtime a été compilé manuellement.
- `compile_docker.sh` en dernier recours. Il compile le projet en un environnement docker où antlr4 et ses dépendances sont bien installés..

2. Lancer le compilateur sur un fichier cible

Pour utiliser le compilateur il faut tout d'abord se rendre dans le dossier: `compiler`.
`cd compiler`
Puis il faut compiler notre compilateur. Pour ceci il suffit de faire un simple:
`make`

Utiliser le compilateur ifcc

Une fois le compilateur crée, son utilisation est relativement simple :
`./ifcc fichier.c`
Cette commande génère le fichier assembleur fichier.s qui pourra être utilisé plus tard pour créer un exécutable (en utilisant gcc par exemple)

Nous avons implémenté certains scripts qui facilitent l'utilisation:

Execute

Le script `execute.sh` utilise notre compilateur pour compiler un fichier nommé `ret42.c`. De plus il compilera le fichier `ret42.s`, qui a été créé par notre compilateur, avec gcc.
Finalement la valeur retournée par la compilation sera affichée dans le terminal.
`./execute.sh`

Run_gcc

Ce script compile simplement le fichier `ret42.c` avec gcc et affiche son résultat dans le terminal.
`./run_gcc.sh`

Test

Ce script (`test.sh`) permet d'afficher les résultats des deux scripts précédents, afin de pouvoir comparer les deux compilateurs plus facilement.
`/test.sh`

Environnement de test

L'environnement de test se trouve sous: /test.
Le fichier `test.sh` à executer avec docker se nomme test.sh et s'execute sans entrée.
Il lance les scripts python `pld-test.py` qui va effectuer les tests en comparant les résultats du compilateur crée avec le compilateur gcc en utilisant `pld_wrapper.sh` :
On le lance simplement ainsi :
`./test.sh`

Le fichier `test_if.sh` lance le script python `pld-test.py` sans passer par docker et supprime le dossier de sortie `pld_test_output`

Le fichier `test_if_log.sh` fonctionne comme `test_if.sh` sauf qu'il conserve `pld_test_output`

3.Résultat de la compilation

Voici le résultat d'une compilation effectuée avec le script test (dans le dossier `compiler`).

```
c3dric@pc-c3dric ..piler-project/SqueletteProjet/compiler (git)-[feature/readme] % ./test.sh
padawan:
10
gcc:
10
```

Fonctionnalités supportées

Un exemple d'un programme supporté par notre compilateur:

```
char f(int i, char c) {
   i = i + 1;
   return c + i;
}

int g(int j) {
   int b =0;
   int d = 4;
   int a =1;
   while (a == 1) {
       if (b == 0) {
           a = 2;
       }
       while (b < 3) {
           int c = 1;
           b = b + c;
       }
       j = b + d;
   }
   return j;
}

char h(int i, char c) {
   c = f(i, c);
   int d = g(c);
   return c + i + 3;
}


int main() {
   int a = 1;
   char b = 'f' + 2 + 'a';
   b = h(a, b);
   return b;
}
```

Main

Seuls les programmes contenant un seul main sans paramètres sont acceptés.

Types

Les seuls types supportés sont les entiers à 32 bits nommés *int* et les caractères de la table ASCII nommés **char**

Expressions

Toutes les expressions arithmétiques et binaires sont autorisées. Il est également possible d'écrire des expressions parenthésées et mixtes. Il est possible d'utiliser aussi les opérateurs de comparaison.

Tableaux

Seul les tableaux à une dimension avec taille constante fixée à la déclaration sont autorisés. L'accès aux cases du tableau est uniquement possible avec un indice constant (par ex, t[1] est possible mais t[i] ne l'est pas).

Structures conditionnelles et boucles

Seul les if, if/else, while et for sont supportées.

Procédures

Toute procédure simple, imbriquée et récursive est acceptée à condition qu'elle soit définie.

Fonctions

Toute fonctions simple, imbriquée et récursive est acceptée à condition qu'elle soit définie.

Entrées/Sorties clavier

Les fonctions getchar et putchar sont supportées.

Les commentaires

Les commentaires avec // et /**/ sont supportés.