Proiect laborator

Folosind utilitarele flex & bison, implementati un interpreter pentru un limbaj de programare asemanator cu C, care indeplineste urmatoarele cerinte:

1. **Tipuri de Date Suportate**:
   - Interpretorul trebuie să suporte tipurile de date **int**, **double** și **float**.
   - Trebuie să permită declararea și inițializarea variabilelor pentru fiecare dintre aceste tipuri.
   - Exemplu: **int a = 5; double b = 3.14; float c = 2.5f;**
2. **Operații Aritmetice**:
   - Implementează operații aritmetice de bază: adunare, scădere, înmulțire, împărțire.
   - Asigură-te că operațiile respectă regulile de conversie a tipurilor în C.
   - Exemplu: **a + b; b * c; a / 2;**
3. **Instrucțiuni de Control**:
   - Implementează instrucțiuni de control: **if**, **else**, **while**.
   - Permite instrucțiuni condiționale și bucle bazate pe condiții logice și comparații.
   - Exemplu:

```
if (a > 3) {
   a = a + 1;
} else {
   a = a - 1;
}
while (a < 10) {
   a = a + 1;
}
```

4. **Input și Output**:
   - Permite citirea de input de la utilizator pentru variabile.
   - Implementează afișarea de output în consolă.
   - Exemplu:

```
printf("Introdu un numar: ");
scanf("%d", &a);
printf("Ai introdus: %d", a);
```

5. **Gestionarea Erorilor**:
   - Tratează erorile comune, cum ar fi împărțirea la zero și depășirea limitelor tipurilor de date.
   - Afișează mesaje de eroare adecvate pentru utilizator.
   - Exemplu:

```
// Împărțirea la zero
int d = 0;
if (d == 0) {
   printf("Eroare: Împărțire la zero!");
} else {
   a = a / d; }
```

6. **Blocuri de Cod**:
   - Permite utilizarea blocurilor de cod, delimitate de acolade **{}**.
   - Asigură gestionarea corectă a scopului variabilelor în interiorul blocurilor.
   - Exemplu:

```
{
  int x = 10;
  printf("%d", x);
}
// x nu mai este accesibil aici
```

7. **Funcții**:
   - Implementează posibilitatea de a defini funcții simple, fără parametri sau cu parametri.
   - Permite returnarea de valori din funcții.
   - Exemplu:

```
int suma(int x, int y) {
  return x + y;
}
int rezultat = suma(5, 3);
```

8. **Conversii de Tip**:
   - Implementează conversii explicite între **int**, **double** și **float**.
   - Exemplu: **double e = (double)a; float f = (float)e;**
9. **Comentarii**:
   - Permite utilizarea comentariilor de o linie (//) și pe mai multe linii (/* ... */).
   - Exemplu:

```
// Aceasta este o linie de comentariu
int x = 5; /* Acesta este un comentariu
pe mai multe linii */
```

10. **Interfață Utilizator**:
    - Dezvoltă o interfață simplă de linie de comandă pentru interacțiunea cu interpretorul.
    - Asigură că interpretorul poate executa atât comenzi individuale, cât și scripturi din fișiere.
    - Exemplu:

În linia de comandă: > int x = 5;
Executarea unui script dintr-un fișier: > run script.txt

Prezentare: Laboratoarele 13 si 14

Proiectul este individual.

Praguri:

1. **Nota 6:** Implementarea punctelor 1,2,4,5,8, 10
2. **Nota 8:** Implementarea punctelor 1,2,3,4,5,6,8, 10
3. **Nota 10:** Implementarea punctelor 1-10