Wprowadzenie do C

Krótki wstęp - główne różnice między C/C++ a Python

Na poprzednich laboratoriach mieliście Państwo wprowadzenie do języka Python. C się znacząco od niego różni. Generalnie naturalnym jest, że poszczególne języki będą się różnić składnią czy słowami kluczowymi, i tak np. w C/C++ zamiast wydzielania sekcji kodu tabami używa się nawiasów klamrowych a na końcu linii daje średniki.

Zasadniczo poza tymi różnicami składniowymi występują jednak trzy fundamentalne różnice między Python a C/C++:

- w Python kod się zwykle interpretuje, tzn. jakiś program czyta Wasz kod i go sobie wykonuje, w efekcie można napisać program, który tylko częściowo ma sens i do momentu jak nie wrzucimy "bełkotu" program się wykona. W C/C++ program jest kompilowany, tzn. całość kodu musi być "poprawna składniowo" żeby kompilator przetłumaczył program w język maszynowy (instrukcje zrozumiałe dla komputera). Warto zwrócić uwagę, że proces kompilacji nadal nie gwarantuje, że program jest wolny od błędów. Przypomina to trochę sprawdzanie pisowni w edytorach tekstu - świetnie wyłapują one literówki ale nie wyłapują zdań, które nie maja sensu.
- W Pythonie typy były "dynamiczne" pisaliśmy x =5 a potem x="nazwa" i kod bez problemu działał, w C zmienna nie może być sobie raz liczbą całkowitą, raz ciągiem znaków a kompilator wymaga podania jej typu (C nie zgaduje czy x=5 to liczba całkowita, czy zmiennoprzecinkowa - po prostu kompilator sypie błędem.
- W C/C++ mamy dużo większą kontrolę nad tym, co kod robi z pamięcią. Tutaj możemy sami zarządzać ile pamięci sobie alokujemy. Plusem takiego rozwiązania jest to, że można napisać bardzo wydajny kod, minusem - zarządzać pamięcią trzeba umieć. Jedną z głównych przyczyn wykrzaczania się aplikacji są błędy w zarządzaniu pamięcią, błędy typu null-cośtam znane z Javowych aplikacji to jeden z wielu przykładów złego zarządzania pamięcią.

Zadanie

Zadanie polega na napisaniu, skompilowaniu i uruchomieniu prostego kalkulatora który:

1. w pętli pyta jaką operację ma wykonać (+,- lub exit)
2. jeśli użytkownik wpisze "exit" program się kończy
3. jeśli użytkownik wpisze + lub - program pyta o dwie liczby, w zależności od znaku wykonuje funkcję dodaj lub odejmij (to powinny być zewnętrzne funkcje przyjmujące 2 argumenty i zwracające wynik)

4. następnie program wypisuje wynik i wraca do punktu 1

Porady praktyczne:

W przeciwieństwie do zwykłych skryptów w Python, które się wykonywały "od góry" (chyba, że coś było zawinięte w funkcje) w C/C++ programy zaczynają się od funkcji main. To main wywołuje ewentualnie inne funkcje. W C++ nie ma luźnego kodu w stylu print("hello") w pierwszej linii kodu.

Przykładowy program wywołujący funkcję:

```

#include <stdio.h> // biblioteka ktora tzeba zaladowac zeby dzialalo printf
void drukuj(){// void na poczatku - funkcja nic nie zwraca
   printf("echo");
}
int main(){ // int funkcja zwraca liczbe calkowita
   drukuj();// wywolaj drukuj
   return 0;
}
```

Wczytywanie jest bardziej skomplikowane. Wygląda inaczej gdy chcemy wczytać jedna wartość (liczba zmiennoprzecinkowa, całkowita, znak) a inaczej gdy chcemy wczytać ciąg znaków. Poniżej przykład wczytywania ciągu znaków.

```
char buf[100];//alokujemy ciag znakow do wczytania customowej komendy
if (scanf("%99s", buf) != 1) {//wczytanie i sprawdzenie czy sie powiodlo
            return; // błąd wejścia
}
if (strcmp(buf, "zakoncz") == 0) {//porownanie czy w buf jest "zakoncz"
            printf("Koniec programu.\n");//
}
```

Czemu kod wygląda tak a nie inaczej?:

1. na początku deklarujemy sobie tablicę znaków o rozmiarze 100 w której będziemy czytać to co wpisze użytkownik, rozmiar powinien być taki, żeby przyjąć całą komendę

2. następnie próbujemy wczytać to co użytkownik wpisał, flaga %99s oznacza wczytanie 99 znaków - tak, żeby na koniec wstawić znak końca ciągu znaków

3. w drugim if-ie porównujemy czy to co wpisał równa się słowu zakończ

A tak wygląda wczytanie liczby całkowitej:

```

int x;//tworzymy zmienna x, podajemy od razu jej typ
printf("Podaj liczbe: ");//pytamy o liczbe
scanf("%d", &x);//flaga %d oznacza "wczytaj liczbe calkowita"
```

Program działa tu następująco:

1. tworzymy zmienną typu całkowitego (int) - jak widać w C już na początku określić trzeba typ
2. pytamy o liczbę
3. wczytujemy liczbę, %d to znak informujący że będziemy wczytywać liczbę całkowitą, &x podaje adres zmiennej x (o adresach będzie więcej na kolejnych zajęciach)

Uruchomienie programu odbywa się w następujący sposób:

1. kompilujemy program (gcc -o nazwa_skompilowanego_programu kod_źródłowy
2. uruchamiany poprzez ./nazwa_skompilowanego programu