Dokumentacja projektu - kalkulatora - programu konsolowego

Ernest Mołczan

15 Listopada 2022

1 OPIS PROJEKTU

Zaprojektować kalkulator konsolowy wykonujący działania: [*dodawanie, mnożenie, dzielenie, dzielenie modulo, potęgowanie*] w systemach: [*dwójkowym, trójkowym, czwórkowym, ... , szesnastkowym*] oraz konwertujący liczby między systemami: [*dwójkowym, trójkowym, czwórkowym, ... , szesnastkowym*].

Kalkulator po uruchomieniu ma czytać dane wejściowe z pliku *me-01-in.txt* i wpisywać dane wyjściowe do pliku *me-01-out.txt*.

2 OPIS ROZWIĄZANIA

2.1 Ogólny opis rozwiązania

Pracę nad swoim kalkulatorem zaczynałem z małą umiejętnością posługiwania się językiem C stąd kod źródłowy w wielu miejscach może być nieoptymalny i niezgodny ze standardem ANSI C. Są to problemy, których jestem świadomy, ale by oddać projekt na czas zdecydowałem się pójść *na skróty*.

Do tej pory zdefiniowałem następujące działania, tylko w systemie **dziesiętnym**:

- Dodawanie,
- Mnożenie,
- Dzielenie,
- Dzielenie modulo,
- Odejmowanie, gdzie wynik zawsze jest wartością bezwzględną,
- Porównywanie liczb,
- Potęgowanie.

W tym momencie w moim kalkulatorze nie zostały zdefiniowane konwersje systemów i działania w różnych systemach, ale działania, które są już

zdefiniowane pozwolą mi zdefiniować konwersje systemowe w dalszej części projektu, jeśli do takowej zostanę dopuszczony.

Działania: *dodawanie, mnożenie, dzielenie, odejmowanie(bezwzględne),* zdecydowałem się zdefiniować na zasadzie działań pisemnych uczonych w szkole podstawowej. W ten sposób otrzymałem bardzo uniwersalne funkcje, które działają dla wszystkich przypadków (*nie zabezpieczyłem jednak jeszcze programu przed dzieleniem przez zero, i działa ono jak dzielenie przez 1*)

2.2 *Opis poszczególnych plików: main.c oraz pliki nagłówkowe .h*

Poniżej przedstawiam chronologiczną listę podejmowanych przeze mnie działań w kodzie pliku *main.c*.

1. Wczytanie danych z pliku *me-01-in.txt* do tablicy *input* typu *char* o rozmiarze 500.
2. Przydzielenie odpowiednim zmiennym/tablicom wartości z tablicy *input*. Poniżej zmienne/tablice:
   - **char** dzialanie - działanie wykonywane w kalkulatorze,
   - **int** system[ ] - system w jakim wykonujemy działanie,
   - **char** argument1[ ] - argument pierwszy działania,
   - **char** argument2[ ] - argument drugi działania,
   - **int** system1[ ] - system z jakiego konwertujemy,
   - **int** system2[ ] - system do jakiego konwertujemy,
   - **int** system-arg[ ] - argument, który konwertujemy.
3. Określenie rozmiaru tablic tablic *int* liczba1[ ], *int* liczba2[ ], *int* wynik[] oraz inicjalizacja tych tablic (w tym miejscu widzę duży defekt mojego kodu przez jego niezgodność ze standardem *ANSI C* i w dalszej części projektu zamierzam to naprawić),
4. Konwersja tablic z argumentami działania z typu *char* na *int*: *char* argument1[ ] $\rightarrow$ *int* liczba1[ ] , *char* argument2[ ] $\rightarrow$ *int* liczba2[ ]

5. Wybór funkcji działania w zależności od wartośći zmiennej *dzialanie*,

6. Nadpisanie zawartości *me-01-out.txt* nową zawartością z wynikiem działania.

Plik ***adding10.h***

W pliku *adding10.h* zdefiniowana jest funkcja

void **dodawanie**(*int* liczba1[ ], *int* liczba2[ ], *int* wynik, *int* m, *int* n, *int* l)

, gdzie *liczba1, liczba2* to tablice z argumentami, *wynik* to tablica na wynik, *m, n, l* to rozmiary każdej z tych tablic.

Funkcja *dodawanie()* dodaje pisemnie tablice *liczba1* i *liczba2* i zapisuje wynik w tablicy *wynik*

Plik ***multi10.h***

W pliku *multi10.h* zdefiniowana jest funkcja

void **mnozenie**(*int* liczba1[ ], *int* liczba2[ ], *int* wynik, *int* m, *int* n, *int* l)

, gdzie *liczba1, liczba2* to tablice z argumentami, *wynik* to tablica na wynik, *m, n, l* to rozmiary każdej z tych tablic.

Funkcja *mnozenie()* mnoży pisemnie tablice *liczba1* i *liczba2* i zapisuje wynik w tablicy *wynik*

Plik ***divide10v2.h***

W pliku *divide10v2.h* zdefiniowana jest funkcja

void **dzieleniev2**(*int* liczba1[ ], *int* liczba2[ ], *int* wynik, *int* m, *int* n, *int* l)

, gdzie *liczba1, liczba2* to tablice z argumentami, *wynik* to tablica na wynik,

m, n, l to rozmiary każdej z tych tablic.

Funkcja *dzieleniev2()* dzieli pisemnie tablice *liczba1* i *liczba2* i zapisuje wynik w tablicy *wynik*

Plik ***sub10.h***

W pliku *sub10.h* zdefiniowana jest funkcja

void **odejmowanie**(*int* liczba1[ ], *int* liczba2[ ], *int* wynik, *int* m, *int* n, *int* l, *int* c)

, gdzie *liczba1, liczba2* to tablice z argumentami, *wynik* to tablica na wynik, *m, n, l* to rozmiary każdej z tych tablic, a *c* to w założeniu wartość zwracana przez funkcję *porownywanie*.

Funkcja *odejmowanie()* odejmuje pisemnie tablice *liczba1* i *liczba2* i zapisuje wynik w tablicy *wynik*

Plik ***compare10.h***

W pliku *compare10.h* zdefiniowana jest funkcja

int **porownywanie**(*int* liczba1[ ], *int* liczba2[ ], *int* wynik, *int* m, *int* n, *int* l)

, gdzie *liczba1, liczba2* to tablice z argumentami, *wynik* to tablica na wynik, *m, n, l* to rozmiary każdej z tych tablic.

Funkcja *porownywanie()* porównuje tablice *liczba1* i *liczba2*, i zwraca wartość 0,1 lub 2, gdzie 0 oznacza, że liczba1 $>$ liczba2, 1 oznacza liczba1 $<$ liczba2 , 2 oznacza liczba1 $=$ liczba2

Plik ***divide10-mod.h***

W pliku *divide10-mod.h* zdefiniowana jest funkcja

void **dzielenie-mod**(*int* liczba1[ ], *int* liczba2[ ], *int* wynik, *int* m, *int* n, *int* l)

, gdzie *liczba1, liczba2* to tablice z argumentami, *wynik* to tablica na wynik, *m, n, l* to rozmiary każdej z tych tablic.

Funkcja *dzielenie-mod()* dzieli modulo tablicę *liczba1* przez *liczba2* z użyciem funkcji *odejmowanie(), mnozenie(), dzieleniev2()* i zapisuje wynik w tablicy *wynik*

Plik ***power10.h***

W pliku *power10.h* zdefiniowana jest funkcja

void **potegowanie**(*int* liczba1[ ], *int* liczba2[ ], *int* wynik, *int* m, *int* n, *int* l)

, gdzie *liczba1, liczba2* to tablice z argumentami, *wynik* to tablica na wynik, *m, n, l* to rozmiary każdej z tych tablic.

Funkcja *potegowanie()* podnosi tablicę *liczba1* do potęgi *liczba2* z użyciem funkcji *mnozenie()* i *odejmowanie()* i zapisuje wynik w tablicy *wynik*

3 WNIOSKI DOTYCZĄCE WYNIKÓW OBLICZEŃ

Wyniki obliczeń wszystkich zdefiniowanych przeze mnie funkcji są poprawne, więc algorytmy działań napisane przez ze mnie również uważam za poprawnie zdefiniowane i uniwersalne.

4 KRÓTKI OPIS PROGRAMU

Program po uruchomieniu zmienia plik wyjściowy *me-01-out.txt*, gdzie wpisuje wyniki obliczeń jak i całą zawartość pliku wejściowego *me-01-in.txt*.

5 SZCZEGÓŁOWE INSTRUKCJE KOMPILACJI I URUCHOMIENIA PROGRAMU

Do kompilacji programu *ernest-molczan.exec* użyłem kompilatora *clang* przeznaczonego na system operacjny *macOS*.

Aby prawidłowo skompilować program należy umieścić wszystki pliki nagłówkowe z rozszerzeniem *.h* umieścić w jednym folderze z plikiem *main.c*. Następnie w linuxowym/unixowym terminalu dostajemy się do folderu, w którym znajdują się wyżej wymienione pliki i w tym samym folderze wpisujemy komendę:

gcc main.c -o <nazwa programu>

Przy kompilacji występują ostrzeżenia, które również chcę wyeliminować w późniejszych wersjach mojego kalkulatora.

Aby prawidłowo uruchomić program należy umieścić pliki tekstowe *me-01-in.txt* i *me-01-out.txt* w tym samym folderze co plik *ernest-molczan.exec*, następnie z poziomu Linuxowego/Unixowego terminala należy należy udać się do folderu, w którym znajdują się pliki *me-01-in.txt*, *me-01-out.txt* i *ernest-molczan.exec* i uruchomić program komendą: ./ernest-molczan.

Aby zmodyfikować dane wejściowe otwieramy plik *me-01-in.txt* i zmieniamy działanie i argumenty wedle uznania (system musi zostać dziesiątkowy, a konwersja w obecnej wersji programu nie działa). Program uruchamiamy komendą *./ernest-molczan* w dokładnie taki sposób jak ten przeze mnie opisany. Po uruchomieniu i zakończeniu programu wynik zostanie zapisany w pliku *me-01-out.txt*. Aby zobaczyć wynik otwieramy plik *me-01-out.txt*

(jeśli plik jest już otwarty, to zamykamy i otwieramy go ponownie).