M++ Compiler 1.5
ALFA

Autor: Michał *Michzimny* Zimniewicz

Spis treści

1. Opis kompilatora

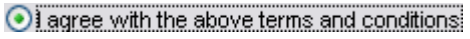
M++ Compiler jest kompilatorem języka M++. Podstawowym składnikiem pakietu jest wsadowy kompilator („mpp.exe”). Nie posiada on własnego edytora, przez co kod programu należy przygotować w dowolnym innym programie zapisującym pliki w formacie TXT (chociażby Notatnik z Windows).

Pliki skompilowane za pomocą M++ Compiler mają format COM, przez co mogą być uruchamiane w dowolnym środowisku kompatybilnym z systemem DOS. Należy pamiętać, że pliki typu COM nie mogą być większe niż 64 kB!

W skład pakietu wchodzi także konwerter plików („com2exe.exe”), który zamienia pliki COM na pliki EXE.

Język posiada bardzo prostą składnię wzorowaną na językach Pascal, C++ oraz Assembler. Wszystkie jego instrukcje są tak zaprojektowane, że prędkością dorównują 32-bitowemu Assemblerowi.

2. Instalacja

Aby rozpocząć proces instalacji uruchamiamy instalatora pakietu (plik „setup_mpp.exe"). Klikamy Next. Pojawi się licencja programu, którą musimy zaakceptować zaznaczając pole 

. Następnie klikamy Next, by przejść dalej. Teraz wybieramy miejsce instalacji programu, a na kolejnym ekranie klikamy Start, aby rozpocząć kopiowanie plików.

Po zakończeniu kopiowania możemy jeszcze przejrzeć listę plików, a następnie proces instalacji kończy się. We skazanym miejscu pojawił się folder „M++ Compiler v1-5a".

Teraz możemy jeszcze skojarzyć pliki *.mpp z edytorem tekstu (np. Notatnik), ale nie jest to koniecznie, bowiem możemy używać również plików *.txt.

3. Opis języka

Poniżej przedstawiam opis składni oraz elementów języka M++. Jest to wstęp do programowania w tym języku. Jeśli kiedykolwiek programowałeś w Pascalu zrozumienie tego języka nie powinno stać się problemem.

3.1. Składnia

Tak wygląda przykładowy program „Hello world”:

```
//program „Hello world”
begin
        put ‘Hello world!!!’
        get
end
```

Program po uruchomieniu wyświetli na ekranie napis „Hello world!!!”, następnie poczeka na wciśnięcie dowolnego klawisza i zakończy się. Linia „//program...” jest komentarzem, który jest ignorowany przez kompilator; ma tylko za zadanie pomagać programiście. **Komentarze rozpoczynamy dwoma slashami - //.**

Begin i End to tzw. „ramy programu”. Wszystkie instrukcje wykonywane przez program muszą się znajdować między nimi. W programie występują tylko raz! Jeśli wywoła się jakąś procedurę przed słowem Begin kompilator zgłosi błąd. Instrukcje znajdujące się za End są przez kompilator ignorowane (po natrafieniu w programie na słowo End kompilator kończy proces kompilacji).

Instrukcja Put jest odpowiedzialna za wyświetlanie tekstu, natomiast Get wstrzymuje program aż do naciśnięcia dowolnego klawisza. Więcej informacji w dziale **Instrukcje**.

Każda instrukcja występuje w osobnej linii!

Parametry podajemy oddzielone spacją od instrukcji (nie jest to jednak koniecznie). Jeśli parametrów jest więcej oddzielamy je przecinkiem. Parametry tekstowe umieszczamy pomiędzy dwoma apostrofami (np. ‘Hello world!!!’).

Wielkość liter w kodzie jest dowolna, ponieważ dla kompilatora „a” i „A” to to samo.

3.2. Instrukcje

Poniżej przedstawione zostały wszystkie instrukcje języka M++ dostępne w wersji 1.5a.

3.2.1. Instrukcje wejścia/wyjścia

Instrukcje służące do wyświetlania tekstu i operacji w trybie tekstowym oraz instrukcje obsługi klawiatury.

Put s Wyświetla na ekranie ciąg znaków s lub zawartość zmiennej typu char. Przykład: put ‘Hello world!!!’ put #ch
Cls Czyści zawartość ekranu. Przykład: cls
Get Zatrzymuje wykonywanie programu i czeka na wciśnięcie dowolnego klawisza. Można dodać jako parametr zmienną char, w której zostanie umieszczony wciśnięty klawisz. Przykład: get get #ch
Nl Przenosi kursor o jedną linię w dół. Przykład: nl
SetTM Uruchamia tryb tekstowy i czyści ekran. Przykład: SetTM //równoznaczne z SetVGA 3

3.2.2. Instrukcje graficzne

Instrukcje służące malowania na ekranie w trybie graficznym $13 (uruchamianym poleceniem SetGM) oraz do jego obsługi.

SetVGA n Ustawia tryb (VGA) pracy karty graficznej. Przykład: SetVGA $13 //tryb graficzny 320x200 (256 kolorów)
SetGM Uruchamia tryb graficzny. Przykład: SetGM //równoznaczne z SetVGA $13
SetSVGA n Uruchamia tryb SVGA (M++ nie posiada instrukcji do obsługi tych trybów). Przykład: SetSVGA $101
BgColor n Zmienia kolor tła na kolor o numerze n. Przykład: BgColor 4 // zmienia kolor na czerwony
Pixel x,y,n Na współrzędnych x,y rysuje piksel o kolorze n. Przykład: Pixel 50, 30, 2 // rysuje zielony piksel (2) na współrzędnych 50, 30

3.2.3. Instrukcje sterujące

Opisy instrukcji skoków oraz pętli.

Loop n ... Stop
Tworzy pętle, która wykonuje instrukcje znajdujące się pomiędzy „Loop” i „Stop” n razy.
Przykład:
Loop 10
  put ‘abc’
  get
Stop //instrukcje put i get zostaną wykonane 10 razy

Jump label
Skacze do etykiety label. Etykieta ta musi wystąpić przed instrukcją Jump, więc pozwala ona tylko na skoki „do tyłu”. Etykiety poprzedzamy dwukropkiem „:”.
Przykład:
:abc
  put ‘jump’
  nl
Jump abc //w nieskończoność przeskakuje do etykiety abc

3.2.4. Pozostałe

Inne instrukcje języka M++.

Nope Nic nie robi. Przykład: Nope
Wait n Zatrzymuje wykonywanie programu na n/10 sekundy. Może działać nieprawidłowo na procesorach szybszych niż 200 MHz. Przykład: wait 20 //czeka 20/10=2 sekundy
Beep Generuje sygnał z głośniczka systemowego. Przykład: Beep
Large n Zwiększa rozmiar programu o n bajtów. Przykład: Large 10 //zwiększ rozmiar programu o 10 bajtów
Huge n Zwiększa rozmiar programu o n kilobajtów. Przykład: Huge 24 //zwiększa rozmiar programu o 24 kB
MCI n Machine Code Instruction. Wstawia instrukcje w kodzie maszynowym. Uwaga! Nieumiejętne użycie w najlepszym wypadku zawiesi program! Przykład: mci $B4 mci $04 // wstaw do kodu instrukcje $B4 $04 = mov ah, 4
Halt Przerywa w tym miejscu działanie programu. Przykład: Halt

3.3. Zmienne

Schemat deklaracji zmiennych wygląda następująco:

typ #nazwa

Zmienne można deklarować w dowolnym miejscu programu (np. przed Begin lub między innymi instrukcjami). Typy dostępne w wersji 1.5a to byte i char. Obydwa maja rozmiar 1 bajtu. Byte przechowuje liczbę w zakresie od 0 to 255, a char znak ASCII. Wartość początkowa zmiennych to 0 (dla typu char będzie to znak NULL – ASCII #0). Przykładowa deklaracja zmiennych:

byte #abc

```
byte #def
char #zxc
char #qwerty
```
Nazwa zmiennej musi się składać z liter, cyfr lub znaku „_”, przy czym pierwszym znakiem zawsze musi być „#” (zarówno przy deklaracji jak i użyciu).

Operacje na zmiennych
```
#abc=x //przypisanie wartości x
#abc++ //inkrementacja (+1)
#abc-- //dekrementacja (-1)
#abc+=x //zwiększenie o x
#abc-=x //zmniejszenie o x
```

W przypadku operacji przypisania, wartość musi być gotową liczbą (nie inną zmienną). Np.
```
#a=18
#a++
#xcv—
#qwerty+=25
#sdf-=212

#abc=#def //BŁĄD!!!
```

3.4. Operatory
Liczba podana jako parametr jakiejś procedury jest traktowana jak liczba dziesiętna. Istnieje jednak możliwość podawania parametrów w innych systemach. Jeśli liczba ma być heksadecymalna (szesnastkowa) to poprzedzamy ją znakiem dolara („$”), a jeśli binarna (dwójkowa) to znakiem procentu („%”).

3.5. Dyrektywy kompilatora
Dyrektywy te sterują procesem kompilacji. Dyrektywy kompilatora poprzedzamy znakiem wykrzyknika „!”. Można je umieścić w dowolnym miejscu programu, ale jako instrukcje, czyli w osobnej linii. Dyrektywy mogą występować przed Begin.

Dyrektywa E – End. Określa kompilatorowi, czy na końcu programu (w miejscu wystąpienia End) program ma się kończyć (tzn. dodawać instrukcje Halt). Jeśli tak się nie stanie, a program korzysta ze zmiennych może się zawiesić.

- !e+ (dodaj instrukcje halt – domyślnie)
- !e- (nie dodawaj tej instrukcji)

4. Przykładowy program

Teraz przedstawię prosty program stworzony w M++.
Najpierw wyświetlimy 3 razy nazwę programu, potem poprosimy użytkownika o wciśnięcie jakiegoś klawisza, a na koniec przejdziemy do trybu graficznego i zmienimy kolor tła.
Kod tego programu dostępny jest w folderze examples pod nazwą example.mpp.

4.1. Kod programu

```
begin
	loop 3 //3 razy powótrz instrukcje put i nl
	put 'Example for M++'
	nl
	stop
	nl

	put 'Wcisnij dowolny klawisz...'
	char #ch
	get #ch //pobierz znak
	nl
	put 'Nacisnales klawisz: '
	put #ch //wyświetl znak

	nl
	nl
	put 'Nacisnij dowolny klawisz, to przejde do trybu graficznego'
	get

	SetGM //uruchom tryb graficzny 13h
	BgColor 2
	get
	BgColor 3
	get
	BgColor 4
	get
end
```

4.2. Kompilacja

Teraz uruchamiamy M++ Compiler (mpp.exe) i podajemy nazwę pliku z kodem źródłowym (lub ścieżkę, gdy jest w innym folderze). W systemie Windows możemy też przeciągnąć ikonę pliku na ikonę mpp.exe. Po podaniu ścieżki wciskamy ENTER. Gdy kompilacja się zakończy pojawi się odpowiedni komunikat. Jeśli w kodzie są błędy wyświetli się komunikat o błędzie (patrz Rozdział 6). W wyniku kompilacji powstanie plik o tej samej nazwie co kod źródłowy i rozszerzeniu COM.

4.3. Konwertowanie do pliku EXE

Stworzony w ten sposób program możemy przekonwertować to formatu EXE. Aby to zrobić należy uruchomić program Com2Exe (com2exe.exe) i podać nazwę pliku COM (lub ścieżkę; patrz wyżej). Można też przeciągnąć ikonę tak jak w przypadku kompilacji. Po konwersji powstanie plik *.exe.

5. Użyte materiały i technologie

Środowisko programistyczne:

Free Pascal Compiler 1.0.6

Dodatkowe narzędzia:

ClickTeam InstallCreator
Netwide Assembler

Materiały:

Ralf Brown's Interrupt List
Intel Architecture Software Developer's Manual

6. Dodatek: Błędy kompilacji

Nr błędu	Błąd	Opis błędu
1.	Nieznane polecenie	Podana instrukcja nie istnieje
2.	Brak parametru	Instrukcja wymaga parametru
3.	Nieprawidłowy parametr	Nieprawidłowy typ parametru
4.	Oczekiwano znaku „:”	Wywołano MCI bez znaku „:”
5.	Oczekiwano „END”	Brak instrukcji END
6.	Oczekiwano „BEGIN”	Brak instrukcji BEGIN
8.	Za długa etykieta	Etykieta może mieć do 8 znaków
9.	Nieznana etykieta	Użyta etykieta nie istnieje
10.	Za dużo etykiet	Maksymalna liczba etykiet to 32
11.	Oczekiwano „LOOP”	Wywołano STOP bez LOOP
12.	Za duża pętla	Pętla może mieć max. 256 bajtów
13.	Oczekiwano „STOP”	Wywołano LOOP bez STOP
50.	Nieprawidłowa deklaracja zmiennej	Nieprawidłowy format deklaracji
51.	Nieznany typ zmiennej	Podany typ nie istnieje
52.	Nieprawidłowa nazwa zmiennej	Użyto nieprawidłowych znaków
53.	Za dużo zmiennych	Max. liczba zmiennych to 128
54.	Nieznana zmienna	Użyto nieznanej zmiennej
55.	Nieprawidłowe użycie zmiennej	Nieprawidłowa składnia
56.	Nieprawidłowa wartość	Wartość poza zakresem
57.	Niezgodność typów	Wartości niezgodna z typem
58.	Niezadeklarowana zmienna	Użyto nieznanej zmiennej
59.	Podwójna deklaracja zmiennej	Dwa razy zadeklarowano zmienną
100.	Podany plik wejściowy nie istnieje	Ścieżka do pliku jest niewłaściwa
101.	Nie można utworzyć pliku *.com	Dysk chroniony przed zapisem
200.	Zbyt długi wiersz	Max. długość wiersza to 120
999.	Nieznany błąd	---