


Wykres - opis działania programu

Ideą tego programu jest doprowadzenie wprowadzonego łańcucha znakowego jako wyrażenia algebraicznego do postaci pozwalającej na systematycznie operacje prostych działań (+, -, *, /) na dwóch liczbach po:

- sprawdzeniu szeregu elementów, które arbitralnie sobie założyłem, takich jak niemożliwość wprowadzenia większej ilości znaków *funkcji dowolnej* niż 70, więcej niż 6 cyfr części całkowitej liczby i tyle samo części ułamkowej, zapisanie w wyrażeniu więcej niż dwukrotnie tej samej funkcji matematycznej (np. $f(x) = \sin(x) + \sin(3 \ln(x/2)) - 1/\sin(x)$ jest niemożliwe dla programu do wykonania ze względu na trzykrotne wystąpienie zapisu *sin*) itd.
- sprawdzeniu czy wprowadzona przez użytkownika liczba jest liczbą i to liczbą rzeczywistą (liczby zespolone nie są akceptowane),
- odkryciu zapisu, który musi być funkcją matematyczną i sprawdzeniu jej składni zgodnej ze składnią podaną w tabeli funkcji, do której jest zawsze dostęp. Np. pierwiastek kwadratowy to *sq()* a nie *sqr()*,
- usunięciu niepotrzebnych spacji wprowadzonych przez użytkownika,
- wprowadzeniu ominiętych w zapisie znaków mnożenia,
- akceptacji tylko dwóch zapisów: 'e' i 'pi' jako stałych liczb; 'pi' zostaje zamienione wizualnie na .
- kontroli nad wprowadzonymi liczbami (granice rozsądku wielkości liczb, składnia ma być z kropką dziesiętną, itd.),
- obliczeniu wartości (dla danego 'x') w nawiasach najbardziej wewnętrznych,
- obliczeniu wartości funkcji matematycznych dla już konkretnej liczby α (lub dwóch konkretnych liczb dla *mod*(α, β) lub *pow*(α, β)), gdy α i β są konkretnymi liczbami.

Kolejność tych ostatnich punktów jak powyżej, wcale nie musi być zachowana.

Program sprawdza czy nawiasy () są ze sobą sparowane ale nie dopisuje brakującego) na końcu wyrażenia.

Zasadnicza większość z tych punktów nie dotyczy 'wielomianu', dla którego też automatycznie podana jest dziedzina funkcji jako $X \in (-\infty; +\infty)$.

Wiadomości wstępne

Ten program nie pretenduje aby być idealnym. Nawet wysyłanie jego kodu do firm poszukujących programistów nie dało mi możliwości, aby być w jednym z nich zatrudnionym. Podobnie było z programem XYZ.bas (trójwymiarowa grafika funkcji - znajdziesz go na tej stronie internetowej) kilka lat wcześniej.

Od 1998 roku, gdy go stworzyłem**, zmieniła się tablica ASCII zarówno angielska jak i polska. Stąd dziwne oznaczenia w programie:

- Między czasie nastąpiło zubożenie znaków matematycznych: Niegdyś dostępny symbol ∞ zamieniłem « lub », \in na =, stąd np. zapis $X = (-\infty; +\infty)$ zamiast poprzedniego $X \in (-\infty; +\infty)$ ***. Zniknął symbol przybliżenia (\sim nad znakiem =) i tylko na angielskiej stronie kodowej pozostał znak \approx .

- Zniknęły znaki alfabetu greckiego: Niegdyś dostępny symbol α zamieniłem na &. Nie ma już znaku π . Angielska strona kodowa jest bogatsza w te znaki. To wynik oszczędności na takich znakach jak polskie:

ą	165		ł	136		ś	152
ć	134		ń	228		ż	171
ę	169		ó	162		ź	190

Ą	164		Ł	157		Ś	151
Ć	143		Ń	227		Ż	141
Ę	168		Ó	224		Ź	189

To aż 18 znaków.

- wymiana numerów znaków graficznych, zarówno w polskim ASCII jak i w angielskim (choć ASCII tych znaków zarówno w polskim jak angielskim ASCII są takie same, ale ich pozycja w tablicy się zmieniła).

A przecież kiedyś na komputerze 'Commodore' definiowałem własne znaki. Podobnie w 'Word Perfect' (poprzednik *Microsoft Word*), zdefiniowałem polskie znaki bez problemu.

Inne mankamenty programu opisałem poniżej.

A jednak często z tego programu korzystałem. Łatwo tu znajdziesz części kodu, które chciałbyś zaimplementować do swoich własnych programów.

Wpisywanie liczb z przecinkiem dziesiętnym zamiast kropką, znajdziesz poniżej: *Wpisywanie liczb z przecinkiem dziesiętnym*.

Trzymałem się standardów takich jak:

- [Esc] dla porzucenia aktywnego obrazu,
- [F1] dla uzyskania informacyjnego obrazu 'pomocy',
- [PgUp] i [PgDn] dla przewijania okien w dół, i z powrotem,
- [Tab] ale jednocześnie strzałki [←] i [→] lub [↑] i [↓] działają we wszystkich wymaganych oknach,
- Jak nie wiesz co robić, po prostu przyciśnij [Enter].

Nie martw się, że podany w notatniku program **wykres.cpp** ma w sobie dziwne znaki zamiast polskich, pomimo, że wpisane do notatnika z opcją 'Kodowanie: UTF-8'. Skopiowany do środowiska Turbo C++ kod powinien pokazywać poprawne znaki.

Podobnie nieistotny jest mogący wystąpić, niekontrolowany rozrzut zapisu (np. większe wcięcia niż zakładano) po skopiowaniu **wykres.cpp** z notatnika do środowiska **Turbo C++**.

* Zapis **sq** jest równie mnemoniczny jak **sqrt**, na który **sq** zostaje w programie zamieniony. W moim założeniu była bowiem możliwość wprowadzenia takich zapisów jak **cosinus(α)** i **kosinus(α)** zamiast **cos(α)**, na który **cosinus** i **kosinus** byłyby zamieniane. Planowałem także zdefiniowanie logarytmu (dodatniej liczby β) o dowolnej, dodatniej podstawie α : **log(α , β)** jako **ln(β)/ln(α)** lub **lg(β)/lg(α)**.

** Taki program próbowałem pisać w LOGO na ZX Spectrum w 1988 r. ale szybko padł on ofiarą małego RAM (16kB).

*** Pozwalam sobie na to, ponieważ w matematyce nie ma ściśle ustalonych standardów. Np. niegdyś przyjęto się oznaczać **ln** jako logarytm naturalny, **lg** jako logarytm dziesiętny, **lb** jako logarytm binarny/dwójkowy i **log** jako logarytm o dowolnej podstawie a jej brak oznaczał podstawę 10. Teraz książki akademickie w ogóle się tego nie trzymają.

Instalacja Borland Turbo C++

Instalacja Turbo C++ jest opisana poniżej na tej stronie internetowej: *Plot (Wykres funkcji) --> Instalacja Borland Turbo C++ i opis programów*.

Funkcje

Funkcje i ich pozycje w programie

Prototypy funkcji i ich linie w programie

void main (void)	234
void wprowadzenie (void);	251
void wybieranie_postaci_funkcji (void);	386
void okno_najwieksze (void);	2212
void wybieranie_wielomianu (void);	453
void usun_spacje_i_analizuj (void);	631
void popraw_zapis_liczby (void);	690
void zbadaj_wielkosc_liczby (void);	737
void wybieranie_funkcji_dowolnej (void);	796
void pytanie_o_zapis (void);	1846
void tablica_operatorow_i_funkcji (void);	894
void grafika_wstepu (void);	1939
void wybory_parametrow (void);	2111
void dlugosc_jednostki_osi_OX (void);	2237

void dlugosc_jednostki_osi_OY (void);	2316
void nr_wzoru_tla_ekranu (void);	2437
void okno_wzorow_tla_i_barw (void);	3092
void nr_koloru_tla_ekranu (void);	2511
void nr_koloru_osi_ukladu_wspolrz (void);	2586
void nr_koloru_wykresu_funkcji (void);	2668
void wstep_rutynowy (void);	2749
void wstepna_analiza_znakow (void);	2782
void brak_mozliwosci_poprawiania (void);	2923
void tlo_wykresu_funkcji (void);	3593
void wykres_funkcji (void);	3750
Raport błędów	
void nie_liczba_rzeczywista (void);	671
void za_dlugi_lancuch (void);	2963
void usun_spacje (void);	2941
void nie_liczba_naturalna (void);	2983
void liczba_poza_zakresem (void);	3003
void kolor_ten_sam (void);	3022
void wyczysc_raport_bledu (void);	3044
void wyczysc_informacje (void);	3070
void znaleziono_blad (void);	4889
Porzucenie programu i pomoc	
void czy_porzucic_program (void);	3162
void zdania_sprzeczne_ze_soba (int ilosc_skokow);	3242
void czy_porzucic_program_gr (void);	3278
void pomoc (void);	3362
void pomoc_gr (void);	3537
Funkcje i ich typy dla "funkcji dowolnej"	
void wpisz_lancuch_i_analizuj_go (void);	991
void popraw_wyrazenie (void);	1812
void czy_jest_argument (void);	1768
void dzialania_w_nawiasie_wewn_test (void);	3989
void dzialania_w_nawiasie_wewn (void);	4414
void usun_tylko_nawiasy (void);	4386
void wprowadz_liczbe_do_lancucha (void);	4314
float wynik_prostego_dzialania (float aa, float bb);	4486
Inne podstawowe (oprócz wpisz_lancuch_i_analizuj_go) elementy "funkcji dowolnej postaci"	
void funkcja_dowolnej_postaci (void);	3960
void oblicz_wartosc_funkcji_matem (void);	4115
int czy_brak_obliczen (void);	4283
Okno "menu" po wykonanym wykresie	
void menu (void);	4522
void analiza_funkcji (void);	4711
void okno_analazy_funkcji (void);	4856

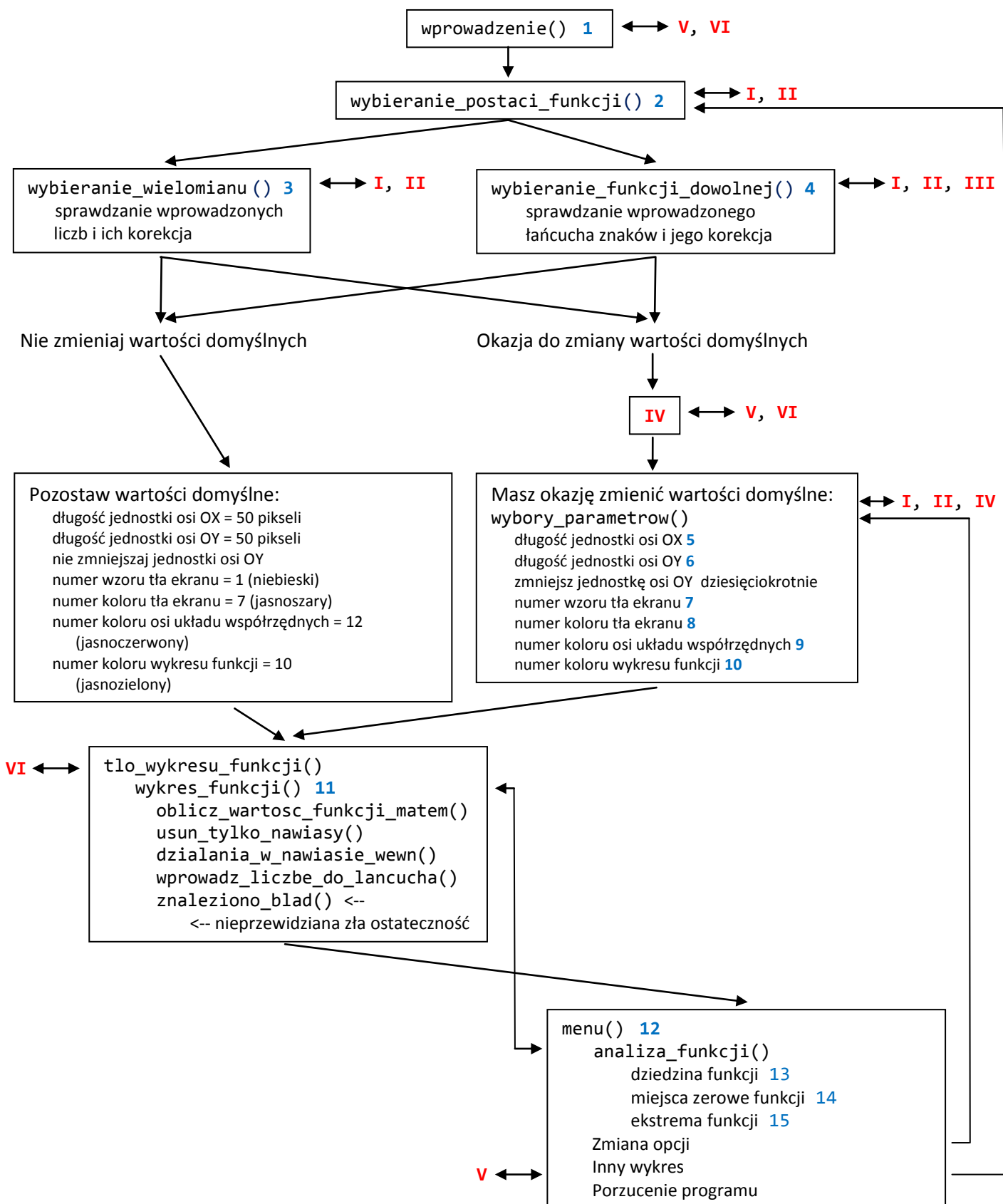
Funkcje - przebieg procesu (flow chart)

Oznaczenia użyte poniżej:

czy_porzucic_program() = I	okno porzucenia programu ekranu niegraficznego
pomoc() = II	informacyjne okno 'pomocy' ekranu niegraficznego
tablica_operatorow_i_funkcji() = III	podgląd zapisu możliwych operatorów i funkcji matematycznych
grafika_wstepu() = IV	lista opcji do zmian z podglądem do tablicy barw
czy_porzucic_program_gr() = V	okno porzucenia programu ekranu graficznego
pomoc_gr() = VI	informacyjne okno 'pomocy' ekranu graficznego

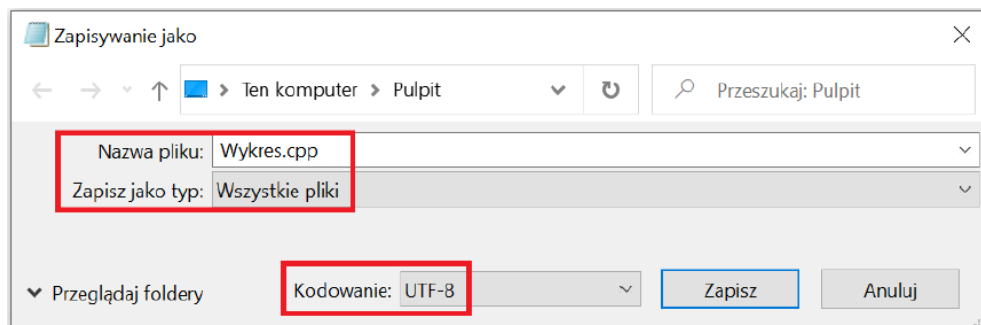
Liczby rzymskie (powyżej) jak i arabskie (poniżej) będą potrzebne aby ponumerować zrzuty ekranów przedstawione pod schematem przebiegu procesu.

Bardzo schematyczny (ale możliwie najbardziej czytelny) przebieg procesu



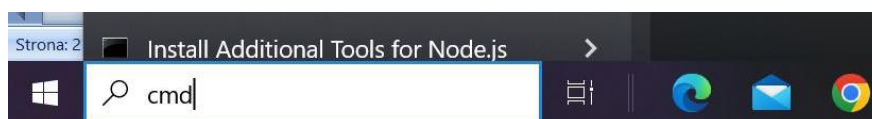
Jak zainstalować program *Wykres.cpp* w środowisku *TurboC3*.

1. Otwórz 'Wykres funkcji' zawierający kod (*Sporządzanie dwuwymiarowego wykresu funkcji - Polish*),
2. Otwórz notatnik Microsoftu,
3. 'copy' całą zawartość 'Wykres funkcji' do notatnika,
4. Zapisz (*Zapisz jako*) na desktop'ie (panelu, pulpitu) plik notatnika z następującymi parametrami:



5. Skopiuj **Wykres.cpp** z pulpitu do **C:\TURBOC3\Projects\Wykres**

a) Otwórz **cmd** (*Command Prompt, Wiersz polecenia*)



b) Zejdź w dół do katalogu głównego (*root directory*) komendą **CD/**

c) W **C:\TURBOC3** utwórz katalog **Projects** a w nim katalog **Wykres**

```
C:\Users\Leszek>cd\  
C:\>  
C:\>cd TurboC3  
C:\TURBOC3>  
C:\TURBOC3>md Projects  
C:\TURBOC3>cd Projects  
C:\TURBOC3\Projects>  
C:\TURBOC3\Projects>md Wykres  
C:\TURBOC3\Projects>cd Wykres  
C:\TURBOC3\Projects\Wykres>
```

d) Skopiuj **Wykres.cpp** z panelu do katalogu **C:\TURBOC3\Projects\Wykres>**

```
C:\TURBOC3\Projects\Wykres>copy C:\Users\ (Twój identyfikator) \Desktop\Wykres.CPP
```

Przykład:

```
C:\TURBOC3\Projects\Wykres>copy C:\Users\Leszek\Desktop\Wykres.CPP
```

Jeżeli będziesz chciał skopiować z powrotem program ze środowiska Turbo C++ na panel, to z dowolnego poziomu w **cmd** wpisz:

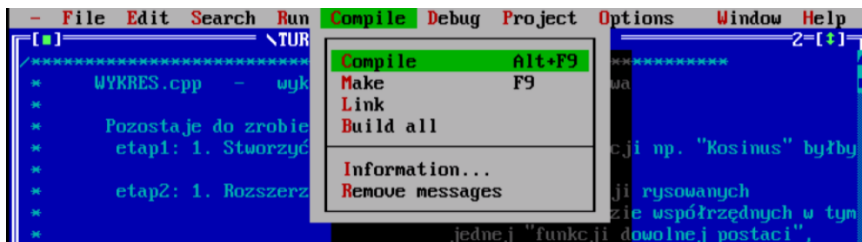
```
copy C:\TURBOC3\Projects\Wykres\Wykres.CPP C:\Users\ (Twój identyfikator) \Desktop\Wykres.CPP
```

Przykład:

```
C:\Users\Leszek>copy C:\TURBOC3\Projects\Wykres\Wykres.CPP C:\Users\Leszek\Desktop\Wykres.CPP
```

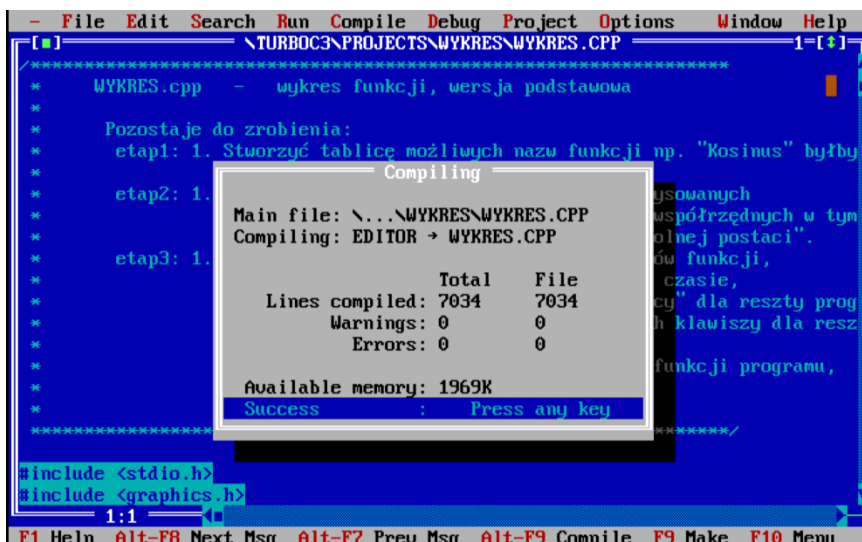
Poprzez program - krok po kroku

Kompilacja programu ([Alt]+C --> [Enter] lub [F10] --> Compile --> Compile --> [Enter] lub [Alt]+[F9]).



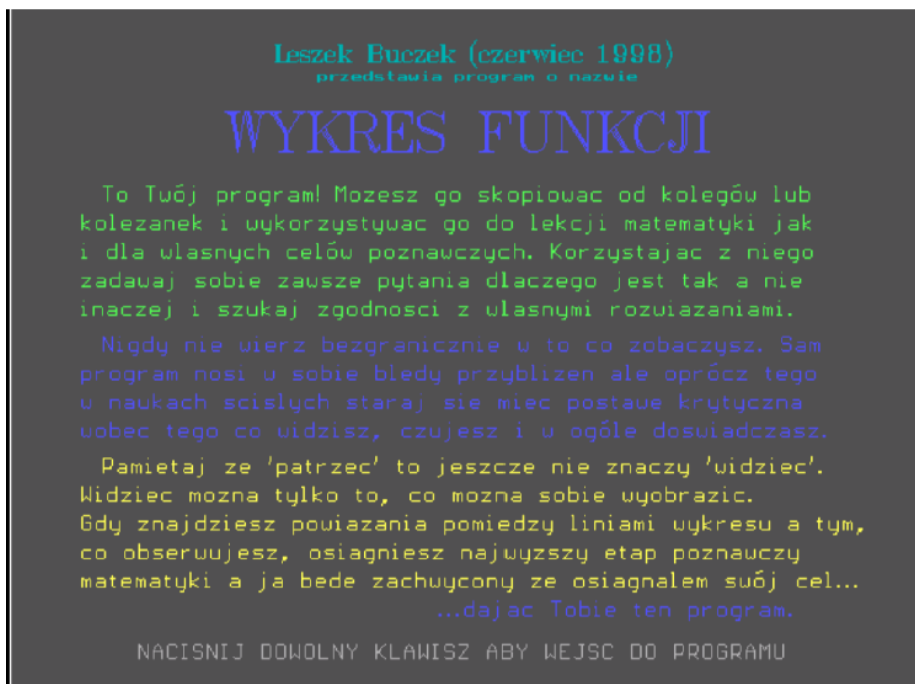
4943 linie programu.

Razem z plikami 'include' - około 7000 linii.



Jeśli masz około 30 błędów kompilacji, prawdopodobnie nie dodałeś do programu „grafiki”. Patrz poniżej: ***Trochę więcej o nawigacji w środowisku Borland Turbo C++ v. 3.0 czyli inne, ważne elementy Menu.***

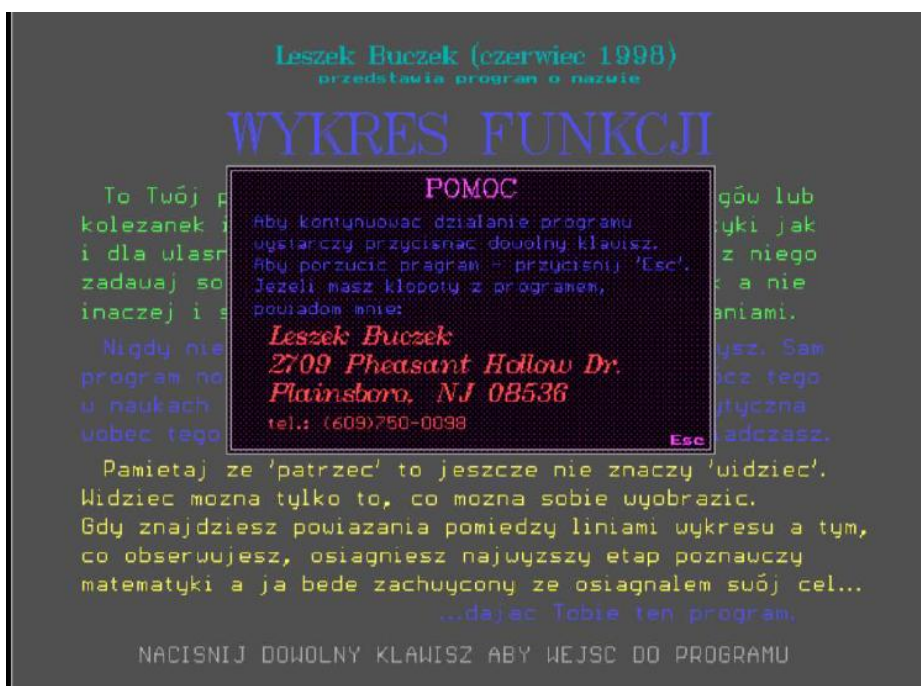
Uruchomienie ([Alt]+R --> [Enter] lub [F10] --> Run --> Run --> [Enter] lub [Ctrl]+[F9]).



Z niemal każdego ekranu są dostępne opcje:

- 'Pomocy' poprzez przyciśnięcie klawisza funkcyjnego [F1], co jest standardem w programowaniu (patrz: funkcja getch()). Wyjście z okna 'Pomocy' następuje z przyciśnięciem [Esc] - też standard.
- 'Porzucenia programu' poprzez przyciśnięcie klawisza [Esc], co jest także standardem w programowaniu.

[F1] (VI) Klawisze operacyjne: [Esc]



[Esc] (V) Klawisze operacyjne: [→], [←], [Tab].



Przeskok pomiędzy 'Nie' i 'Tak' można wykonać albo klawiszami strzałek [→] i [←] albo [Tab].

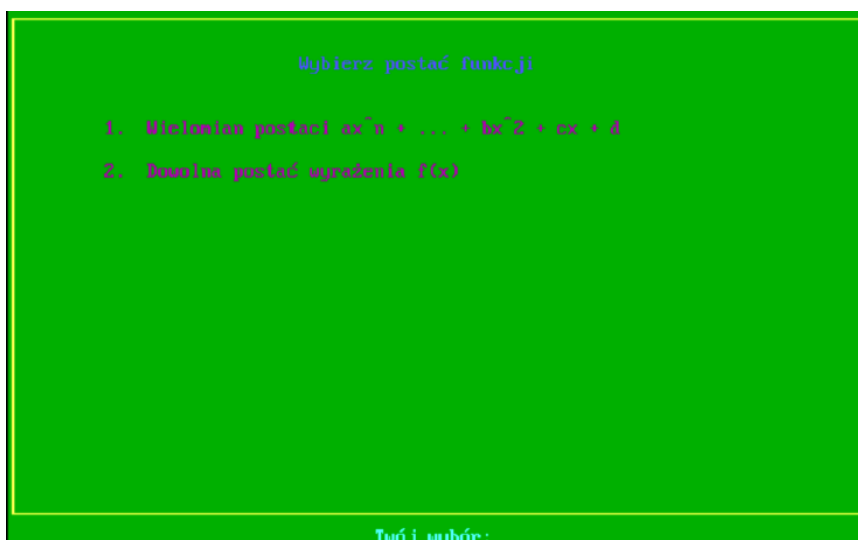
Z okien 'POMOC' wychodzimy klawiszem [Esc].

[Enter] daje wejście do konkretów (jak nie wiesz co robić, przyciskaj [Enter]) podzielonych tu na sekcje:

Sekcje:

1. Wybór rodzaju funkcji: 'Wielomian' czy 'Dowolna postać wyrażenia' (2)
 - a. Postać wielomianowa (3)
 - b. Dowolna postać wyrażenia (4)
2. Zmiana parametrów wykresu funkcji
3. Menu wykresu funkcji
4. Przykład operacji na liczbach stałych: 'e' i 'pi'

Sekcja 1. Wybór rodzaju funkcji: 'Wielomian' czy 'Dowolna postać wyrażenia' (2)

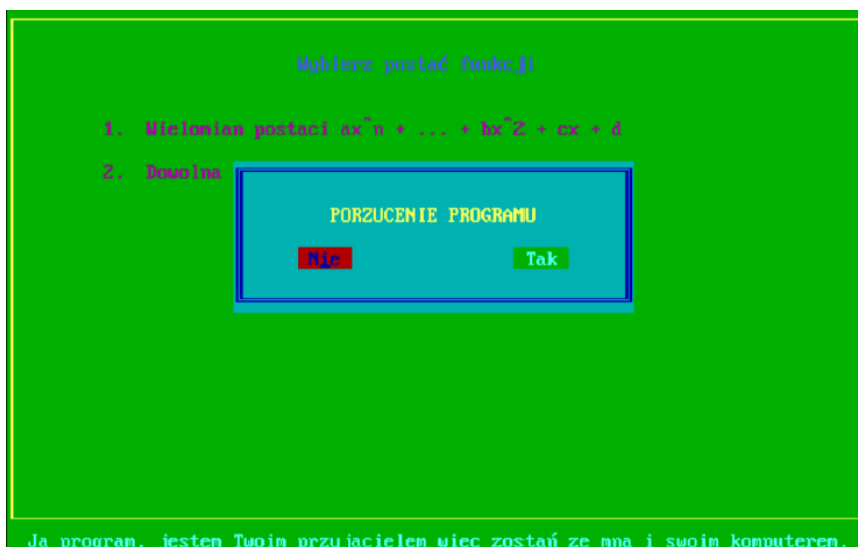


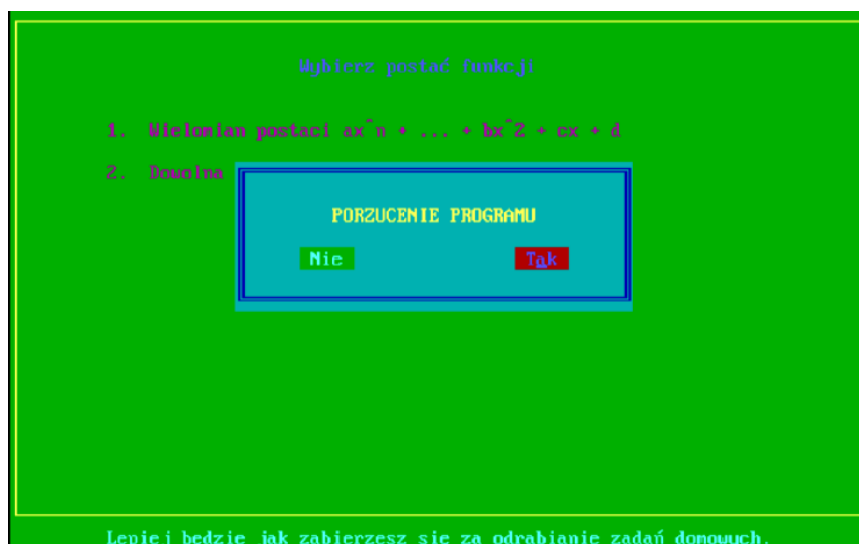
Z tego poziomu można oczywiście wywołać zarówno 'POMOC' - też [F1], tym razem okna niegraficznego - jak i wybrać możliwość 'PORZUCENIE PROGRAMU' - też [Esc], tu jest okno niegraficzne (po wyjściu z obrazu wstępu zamknięta została grafika - komendą `closegraph();`)

'POMOC' okna niegraficznego. Pięć takich okien - operuj klawiszami [PgDn], [PgUp] i [Esc]. (II)



'PORZUCENIE PROGRAMU' okna niegraficznego (I):





Zauważ zmieniające się zdania na dole ekranu. Wydaje się to być niepotrzebne ale...

...każdy program edukacyjny winien być możliwie wysoko interaktywny - to jest tylko przykład z wielu elementów interakcji programu wobec poczynań użytkownika.

Trochę takiej interakcji jest w 'POMOC'y: Jest 6 różnych ekranów/stron 'POMOC'y trybu niegraficznego - odpowiednia strona pojawia się na odpowiednim etapie programu. Jakkolwiek ze wszystkich pojawiających się stron, możliwe jest przewijanie po wszystkich pozostałych stronach.

- - - - -

Wracamy do wyboru:

- albo 'wielomian'
- albo 'funkcja dowolnej postaci'.

I niech to będzie teraz **1** (czyli wielomian). **[Enter]** nie jest tu potrzebny.

Sekcja 1. a. Postać wielomianowa (3)



Jest możliwość wprowadzenia wielomianu do stopnia dziewiątego - tu wpisujemy jego współczynniki w liczbach rzeczywistych. Np.:

Wielomian
 $ax^n + \dots + bx^2 + cx + d$

Podaj stopień wielomianu (1÷9): 6

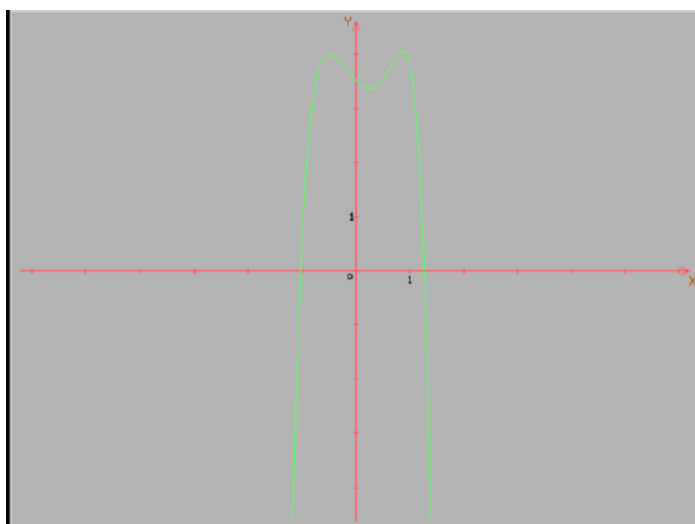
$y(x) = a(1)x^6 + a(2)x^5 + a(3)x^4 + a(4)x^3 + a(5)x^2 + a(6)x + a(7)$

Podaj współczynniki wielomianu 6-go stopnia :

a(1) = -2.45	a(6) = -1
a(2) = 0.6	a(7) = 3.5
a(3) = 0	
a(4) = 2	
a(5) = 1.125	

[Enter] – wykres funkcji; [p] – wybór parametrów wykresu

Po przyciśnięciu **[Enter]**, dostajemy natychmiast wykres funkcji z niezmienionymi jego parametrami opcjonalnymi. **(11)**



Gdy zamiast **[Enter]** przyciśniemy **[p]** to wejdziemy na poziom wyboru szeregu parametrów wykresu funkcji ze wstępnym obrazem jak poniżej (a więc jest możliwość ich zmiany) **(IV)**:

WYKRES FUNKCJI

Będziesz wybierać parametry odpowiadając na następujące pytania:

- Długość jednostki na osi OX: w punktach ekranu (od 10 do 300) [opcjonalnie]
- Długość jednostki na osi OY: w punktach ekranu (od 5 do 220) [opcjonalnie]
- Numer wzoru tła z listy wzorów innego ekranu (od 0 do 11) [opcjonalnie]
- Numer koloru tła z poniższej palety barw (od 0 do 15) [opcjonalnie]
- Numer koloru osi układu współrzędnych (od 0 do 15) [opcjonalnie]
- Numer koloru wykresu funkcji (od 0 do 15) [opcjonalnie]

Do tego obrazu będziesz mieć zawsze podgląd poprzez znak ?

NACISNIJ DOWOLNY KŁAWISZ ABY WEJŚĆ DO PROGRAMU

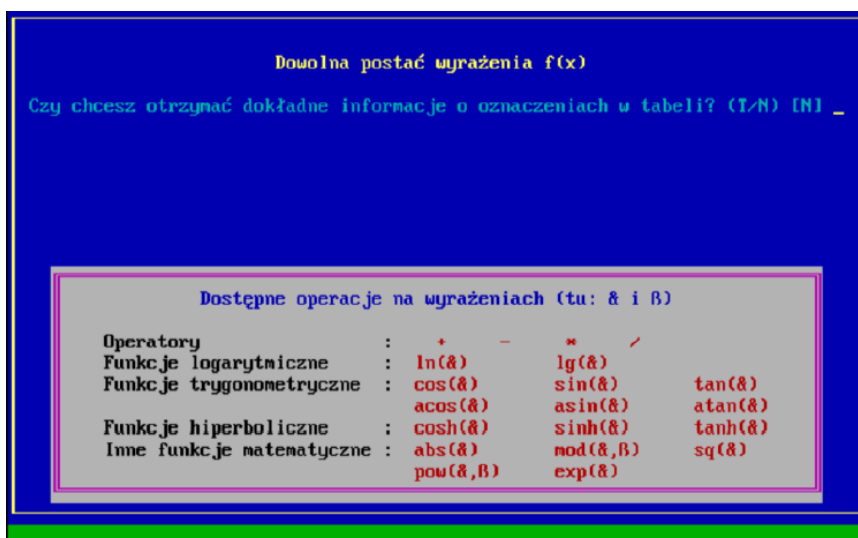
PALETA BARW

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15

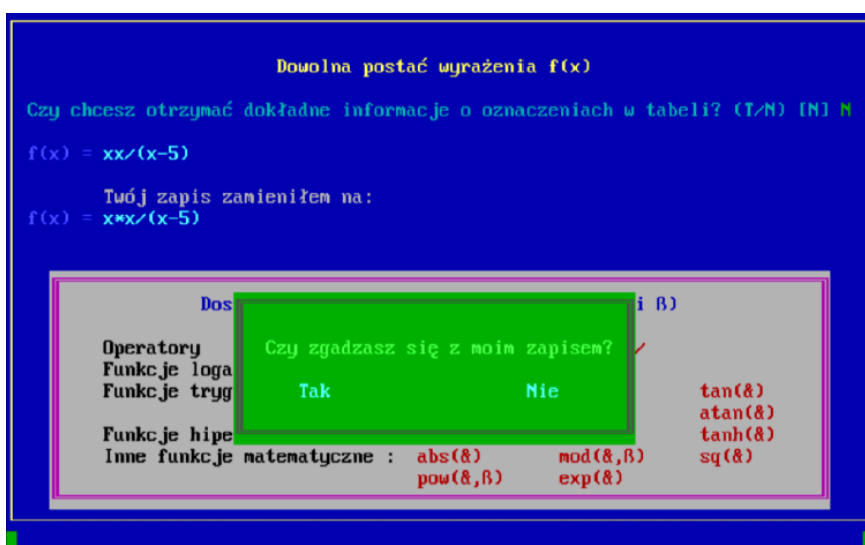
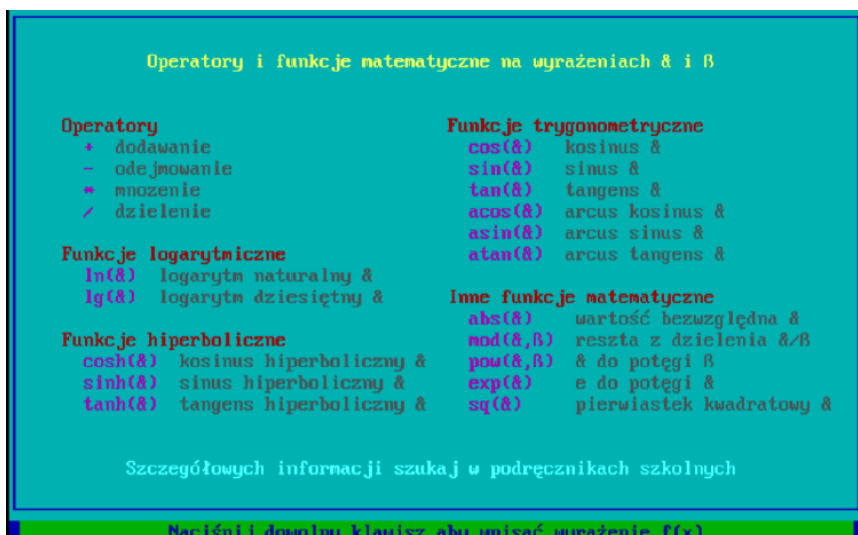
Programy Urszka Burzka

Dalsze wiadomości na temat możliwości zmiany parametrów wykresu, patrz poniżej: **Sekcja 2. Zmiana parametrów wykresu funkcji.**

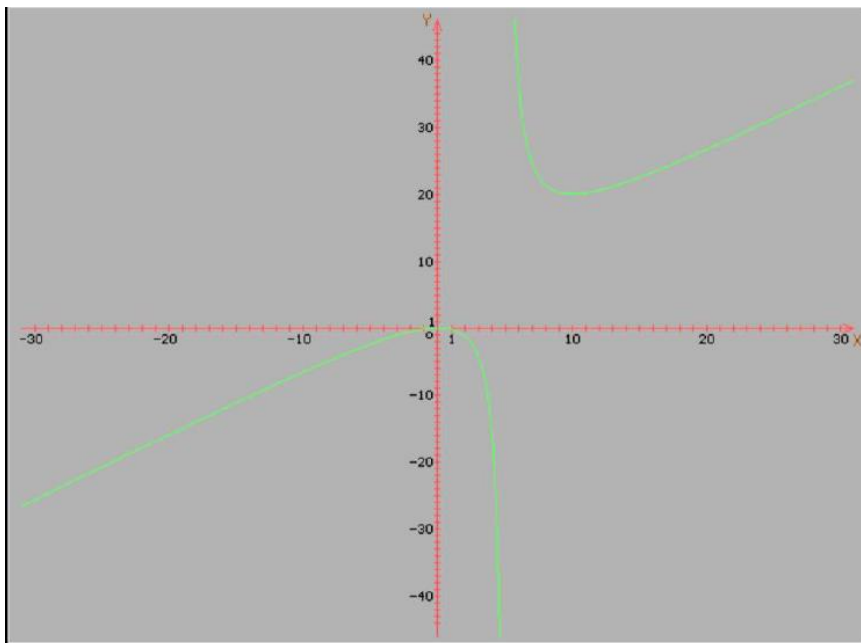
Sekcja 1. b. Dowolna postać wyrażenia (4)



Wstępne zapytanie o szczegółowe zapisy operatorów i funkcji jest opcjonalne: Przeciśnięcie [Enter] omija ten ekran, [t] lub [T] prowadzi do niego:



A po zmianie jednostek długości aby były jak najkrótsze (11):



(13)

```

ANALIZA FUNKCJI
Dziedzina funkcji

X = (?;4.90) u (5.10;?)

Uwaga: '(' może okazać się być '['
a ')' ']' . Pytaniek oznacza że wykres
zaczyna się lub kończy liczbą już
należącą do dziedziny funkcji.
Page Down / Esc

```

(14)

```

ANALIZA FUNKCJI
Miejsca zerowe funkcji

y = 0 dla X1 = 0.00

Page Down / Page Up / Esc

```

(15)

```

ANALIZA FUNKCJI
Ekstrema funkcji

Max: y = -0.00 dla X = 0.00
Min: y = 20.00 dla X = 10.00

Page Up / Esc

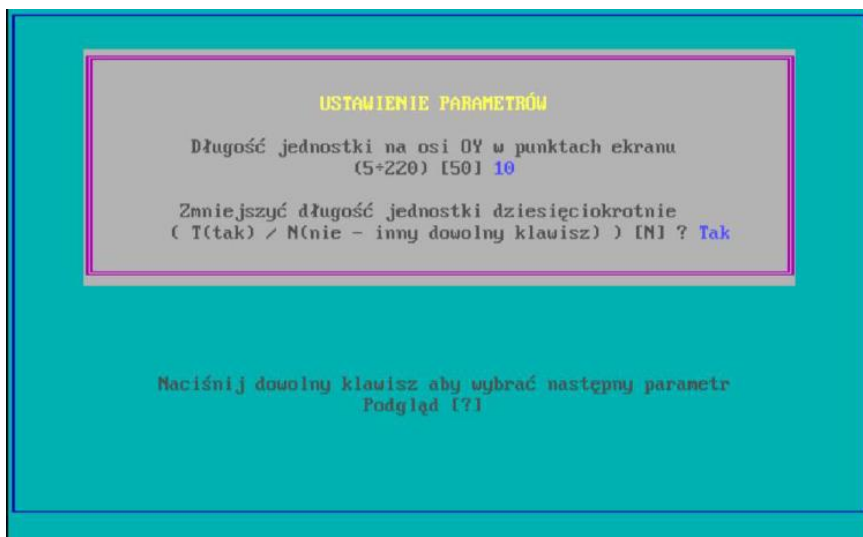
```

Sekcja 2. Zmiana parametrów wykresu funkcji

Można zaakceptować opcjonalne wartości dla każdego parametru przyciskając od razu [Enter] (5) albo ...



... niektóre z nich zmienić (6):

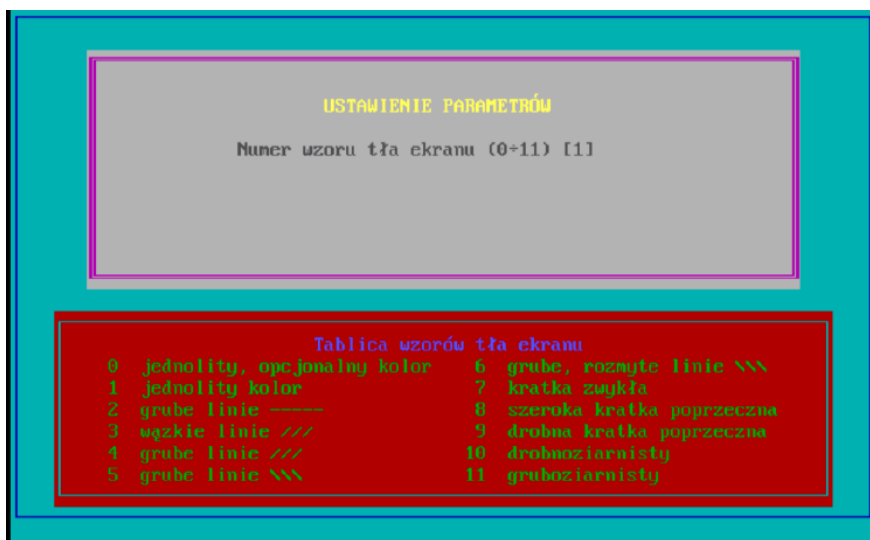


Z każdego takiego ekranu jest podgląd do ekranu głównego zmiany parametrów - **[Shift]+[?]** . Ekran podaje, w którym miejscu zmian się znajdujemy (**IV**):

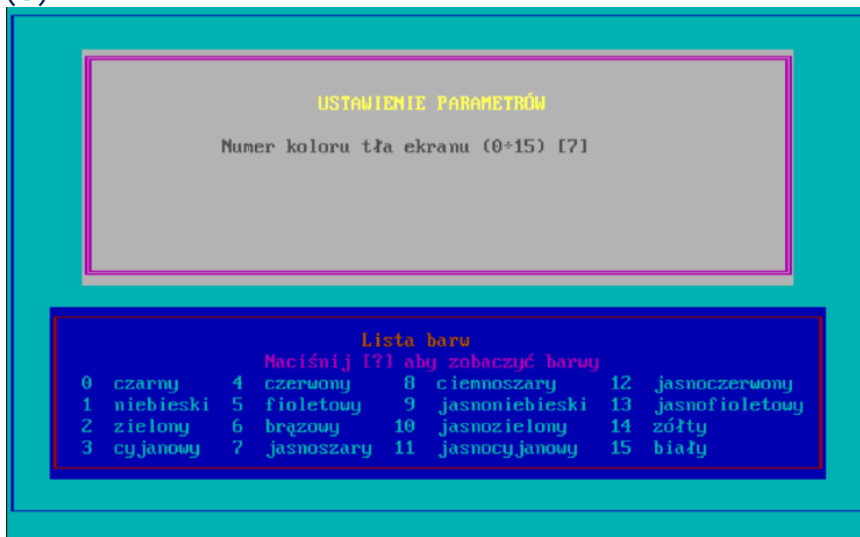


[Enter]

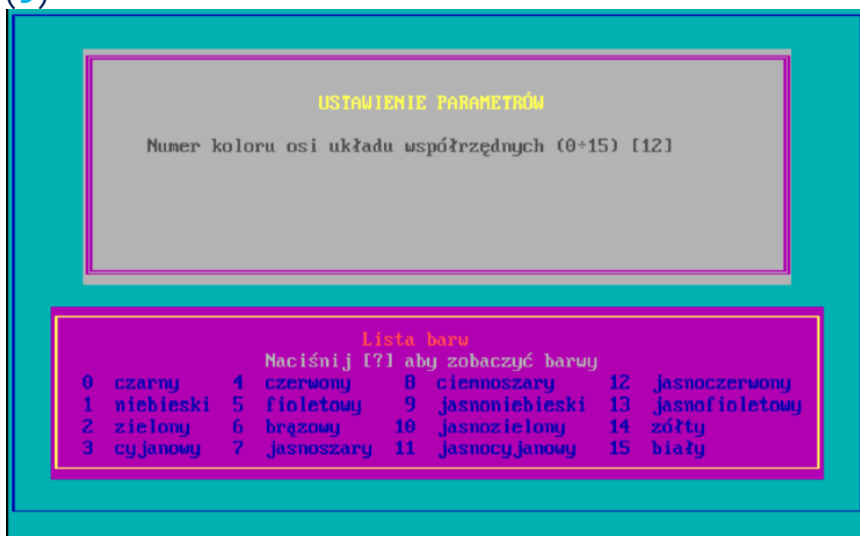
I tak samo z innymi parametrami (**7**):



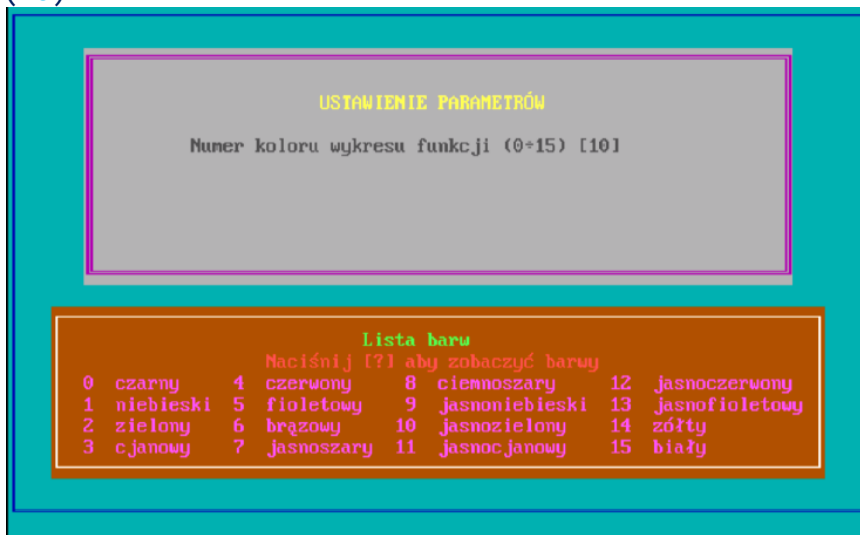
(8)



(9)



(10)

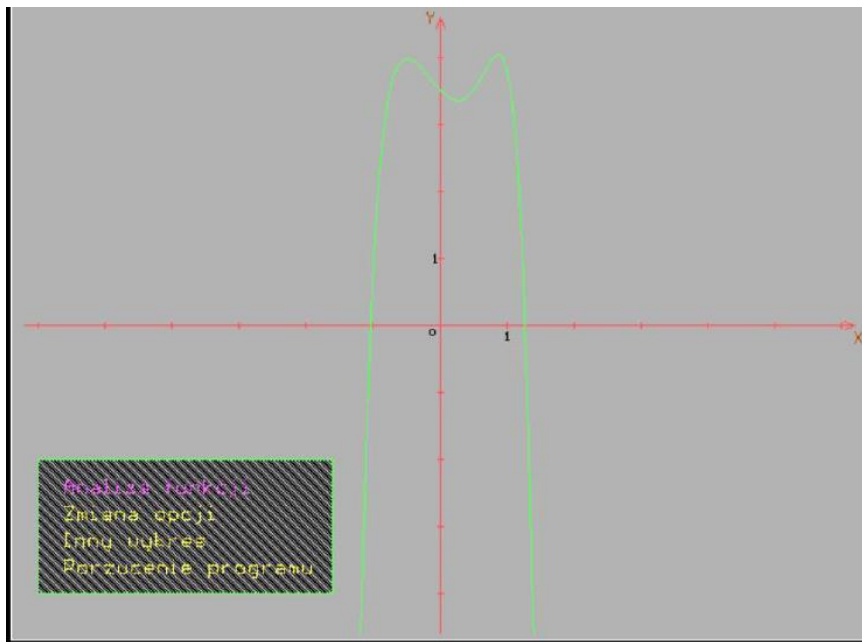


i w końcu:

Naciśnij dowolny klawisz aby zobaczyć wykres funkcji

Sekcja 3. Menu wykresu funkcji

Przyciśnięcie [Enter] na wykresie funkcji uruchamia 'Menu wykresu' (12).



Przeskok pomiędzy tymi czterema opcjami to klawisze [↓] i [↑] a przeskok ten jest 'zwijany' ('**Porzucenie programu**' i [↓] daje znowu '**Analiza funkcji**'), tak jak każda opcja w menu edytora Turbo C++.

Analiza funkcji

Zawiera ona trzy elementy:

- i. Dziedzina funkcji
- ii. Miejsca zerowe funkcji
- iii. Ekstrema funkcji

Dziedzina funkcji dla 'wielomianu' jest automatycznie podana jako $X \in (-\infty; +\infty)$, na ekranie $X = (-\ll; +\gg)$ z wyżej wymienionych powodów.

Pozostałe wyniki są efektem odczytu w trakcie dodatkowych obliczeń prowadzących do sporządzenia wykresu i to tylko dla wartości X i Y mieszczących się na ekranie. Stąd wartości te należy traktować jako przybliżone. Aby uzyskać lepsze przybliżenie tych liczb, należy zmienić opcje sporządzania wykresu:

Długość jednostki na osi OX w punktach ekranu

i/lub

Długość jednostki na osi OY w punktach ekranu

Tak więc dla np. $\sin(x)/x$ otrzymuje się $X = (?; -0.02)$ u $(0.02; ?)$. Znak ? zastępuje tu $-\infty$ i $+\infty$ (w praktyce $-\ll$ i $+\gg$) bo skąd program ma wiedzieć co się dzieje poza ekranem. Zmiana '**długości jednostki na osi OX w punktach ekranu**' z 50 na 300 daje już $X = (?; -0.00)$ u $(0.00; ?)$.

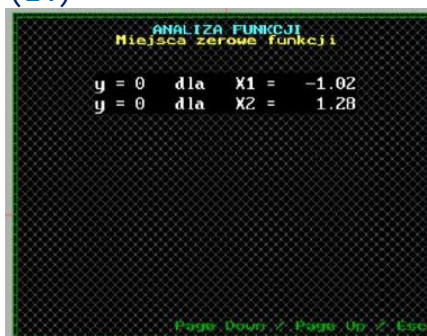
Liczby podawane są do dwóch miejsc po przecinku (tu: kropce).

Dla przypadku wykreślonej funkcji to:

(13)



(14)



(15)



Zmiana opcji



Prowadzi do **Sekcja 2. Zmiana parametrów wykresu funkcji** z tym, że wstępny jego obraz się nie pojawia, ale jest on dostępny z każdego wywoływanego parametru wykresu do ewentualnej zmiany.

Inny wykres

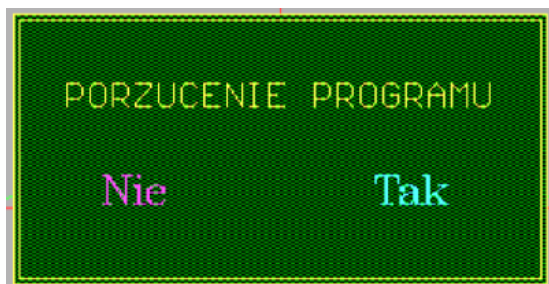


Prowadzi do **Sekcja 1. Wybór rodzaju funkcji: 'Wielomian' czy 'Dowolna postać wyrażenia' ?**

Porzucenie programu



Prowadzi do ekranu, z którego można opuścić program lub anulować porzucanie programu:



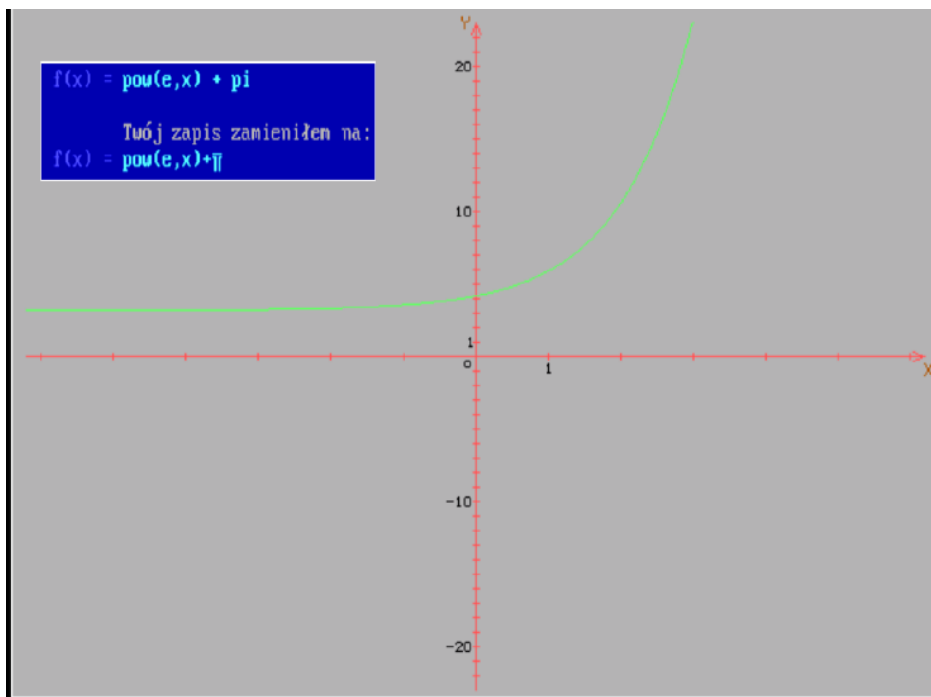
Sekcja 4. Przykład operacji na liczbach stałych: 'e' i 'pi'

Liczbę Eulera można zapisać w wyrażeniu jako **e**. Program zamieni **e** na 2.718282 . Jednocześnie zostawi symbol **e** w wizualnym zapisie wyrażenia.

Liczbę π można zapisać jako **pi**. Program zamieni ten zapis na 3.141593 . Jednocześnie zamieni Twoje **pi** w wizualnym zapisie wyrażenia na graficzny symbol .

Przykład jednoczesnego zapisu liczby Eulera jak i liczby π jest poniżej. Ta funkcja to

$f(x) = e^x + \pi$ Jak widać na wykresie, $f(x) = e^x$ jest przesunięta w górę o 3.14 .



Niektóre elementy działania programu, których możesz nie rozumieć

1. Czekanie na przyciśnięcie jakiegoś klawisza

Z definicji funkcji `getch()`, wykorzystano następujące klawisze, których numery wprowadzono do programu:

```
#define KEY_F1      59
#define KEY_UP      72
#define KEY_PGUP    73
#define KEY_LEFT    75
#define KEY_RIGHT   77
#define KEY_DOWN    80
#define KEY_PGDN    81
```

Oprócz tego z tablicy ASCII:

Enter	13
Esc	27

2. "W pamięci nie ma miejsca dla przechowania obrazu"

Jeżeli uzyskasz błąd...:

Błąd (n): W pamięci nie ma miejsca dla przechowania obrazu
 Jego rozmiar = *wymagana liczba* bajtów jest za duży
 Wolnej pamięci jest tylko *dostępna liczba* bajtów

gdzie,

n - to kolejny numer informacji; pozwala na konkretną lokalizację błędu w kodzie dla podobnych wyświetlanych informacji,

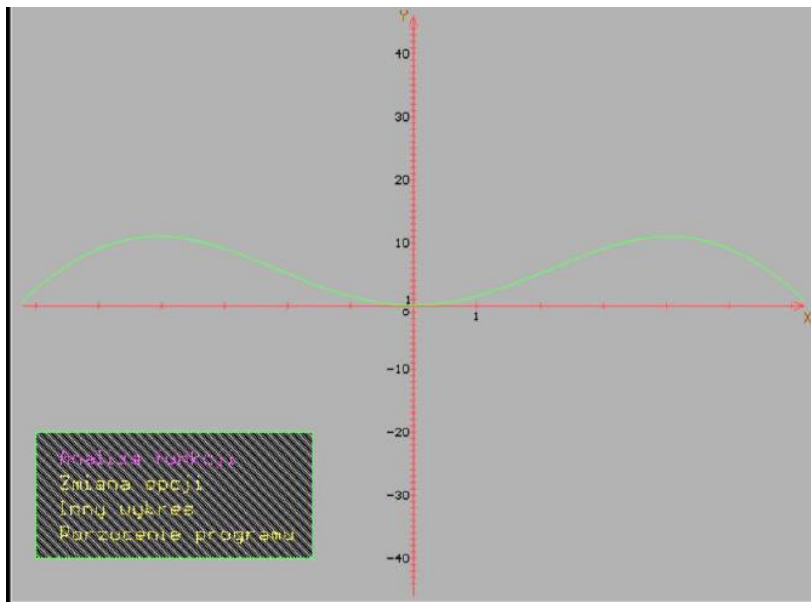
wymagana liczba - to konkretna liczba bajtów wymagana aby obraz mógł się pojawić na ekranie

dostępna liczba - to ilość bajtów w pamięci operacyjnej programu.

Aby obraz mógł się pojawić (a więc cały program mógł działać) musi być spełniony warunek:

wymagana liczba <= *dostępna liczba*

Gdy próba uruchomienia '**Analiza funkcji**'...

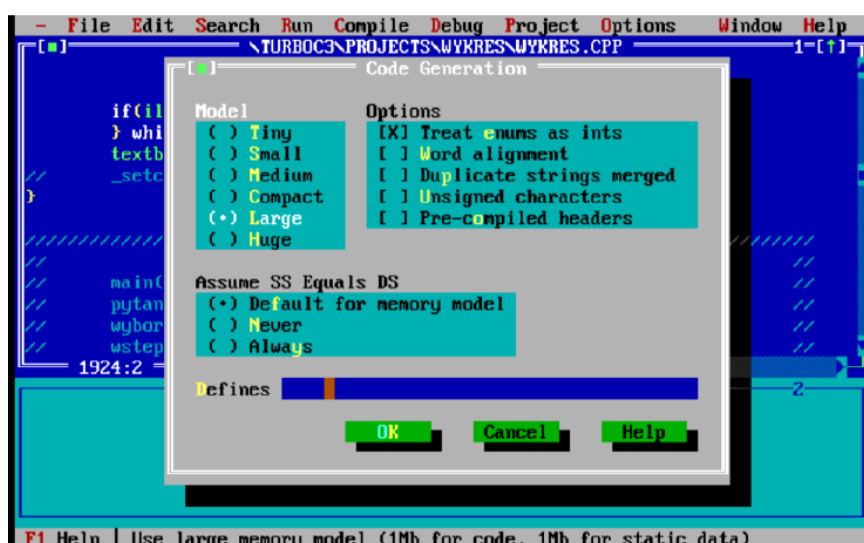
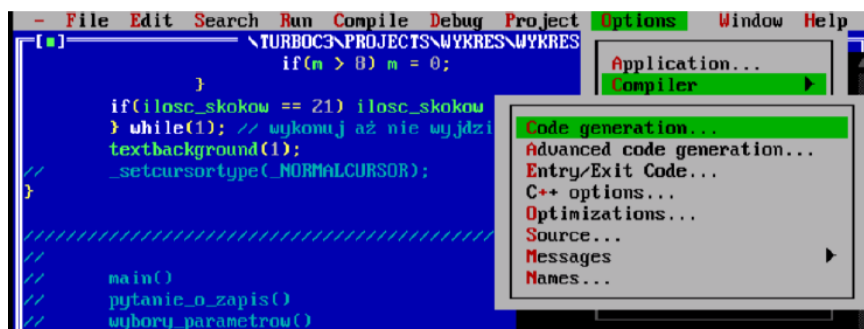


... kończy się błędem:

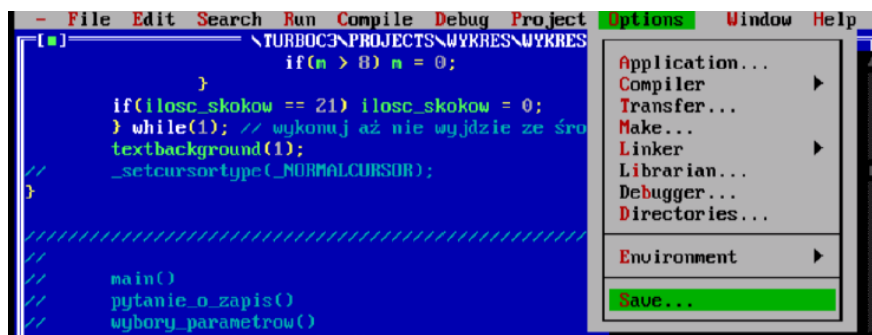
Błąd (?): W pamięci nie ma miejsca dla przechowania obrazu
Jego rozmiar = 44898 bajtów jest za duży
Wolnej pamięci jest tylko 36032 bajtów

To zapewne masz **Model** pamięci zbyt mały. Trzeba go zwiększyć.

Wtedy: **Option** --> **Compiler** --> **Code generation...** --> (zmień 'Model' na 'Large', chociaż 'Compact' tu wystarczy)



Nie zapomnij zapisać tej zmiany:



3. Interaktywne zdania w programie

Za wyjątkiem „POMOCY” (tryb graficzny i niegraficzny) oraz zestawu zdań towarzyszących „WYJŚCIE Z PROGRAMU” (tryb niegraficzny), podczas uruchamiania programu mogą pojawić się następujące wskazówki i/lub ostrzeżenia:

"**Błąd grafiki (moduł 'nazwa modułu, np. wprowadzenie')**: 'kod błędu (errorcode)'"

"**Przyciśnij dowolny klawisz aby wejść do edytora** : "

<-- są 3 różne nazwy modułu a więc i miejsca w kodzie ponieważ grafika jest wywoływana 3 razy

Możliwe informacje pojawiające się u dołu ekranu:

```
"                               ŻLE ! Masz do wyboru "1" albo "2"                               "
```

```
"                               Wybierz jedną z dwóch opcji (1 albo 2)                               "
```

```
"                               ŻLE ! Masz do wyboru cyfry 1, 2, 3, 4, 5, 6, 7, 8 i 9                               "
```

```
"                               Wybierz jedną z dziewięciu opcji (od 1 do 9)                               "
```

```
" Niedozwolona większa ilość znaków. [ENTER] - akceptuj, [<--] - usuń cyfry.                               "
```

```
" Liczba może składać się maksymalnie z 14 znaków. Dopuszczone są spacje.                               "
```

```
" [Enter] - wykres funkcji; [p] - wybór parametrów wykresu                               "
```

```
"                               ŻLE ! Tu ma być liczba rzeczywista w zapisie dziesiętnym.                               "
```

```
" Przykłady: 3, 4.55, .61, 0.61, -.61, -0.61, -123.45, +123.45                               "
```

```
" Nie przyjąłem! Dopuszczalna górna granica liczby to 999999.                               "
```

```
" Już MILION (1000000) to dla mnie liczba za duża.                               "
```

```
" Nie przyjąłem! Dopuszczalna dolna granica liczby to -999999.                               "
```

```
" Liczba MINUS MILION (-1000000) i mniejsze to dla mnie liczby za małe.                               "
```

```
" Uwaga: Ograniczyłem zapis liczby obcinając ją do 6-iu cyfr po kropce.                               "
```

```
" Naciśnij dowolny klawisz aby wpisać wyrażenie f(x) _                               "
```

```
" W tym miejscu wpisz dowolne wyrażenie algebraiczne.                               "
```

```
" Przykład: 4abs(sin(2.5pi/cos(x2tanh(3.2x))))-pow(e,-x/pi)                               "
```

```
"                               Zbyt dużo cyfr w liczbie.                               "
```

```
"Liczba nie może mieć więcej niż 6 cyfr części całkowitej i tyle samo ułamkowej"
```

```
"                               Nie poradzę sobie z większą ilością znaków.                               "
```

```
" Możesz wpisać tylko do 70 znaków. [ENTER] - akceptuj, [<--] - usuń znaki.                               "
```

```
" Stwierdzono istnienie niedopuszczalnego znaku. Popraw zapis wyrażenia.                               "
```

```
" Działanie wymaga dwóch liczb. Popraw zapis wyrażenia.                               "
```

```
" Błąd w zapisie znaków działania. Popraw zapis wyrażenia.                               "
```

```
" Błąd w zapisie nawiasów. Popraw zapis wyrażenia.                               "
```

```
" Funkcja "ln()" może być użyta tylko dwukrotnie. Uwzględnij to w wyrażeniu.                               "
```

```
" Funkcja "lg()" może być użyta tylko dwukrotnie. Uwzględnij to w wyrażeniu.                               "
```

```
" Funkcja "cos()" może być użyta tylko dwukrotnie. Uwzględnij to w wyrażeniu.                               "
```

```
" Funkcja "cosh()" może być użyta tylko dwukrotnie. Uwzględnij to w wyrażeniu.                               "
```

```
" Funkcja "sin()" może być użyta tylko dwukrotnie. Uwzględnij to w wyrażeniu.                               "
```

```
" Funkcja "sinh()" może być użyta tylko dwukrotnie. Uwzględnij to w wyrażeniu.                               "
```


" Funkcja "sq()" może być użyta tylko dwukrotnie. Uwzględnij to w wyrażeniu. "

" Funkcja "tan()" może być użyta tylko dwukrotnie. Uwzględnij to w wyrażeniu. "

" Funkcja "tanh()" może być użyta tylko dwukrotnie. Uwzględnij to w wyrażeniu. "

" Funkcja "acos()" może być użyta tylko dwukrotnie. Uwzględnij to w wyrażeniu. "

" Funkcja "asin()" może być użyta tylko dwukrotnie. Uwzględnij to w wyrażeniu. "

" Funkcja "atan()" może być użyta tylko dwukrotnie. Uwzględnij to w wyrażeniu. "

" Funkcja "abs()" może być użyta tylko dwukrotnie. Uwzględnij to w wyrażeniu. "

" Funkcja "mod()" może być użyta tylko dwukrotnie. Uwzględnij to w wyrażeniu. "

" Funkcja "pow()" może być użyta tylko dwukrotnie. Uwzględnij to w wyrażeniu. "

" Funkcja "exp()" może być użyta tylko dwukrotnie. Uwzględnij to w wyrażeniu. "

" Spodziewana funkcja a jest niepoprawny jej zapis. Napisz poprawne wyrażenie. "

" Coś się nie zgadza z przecinkami. Pamiętaj "." zastępuje "," w liczbie. "

" Każda funkcja wymaga argumentu. Sprawdź w powyższej tabeli i popraw błędy. "

" Funkcje "mod()" i "pow()" są dwuargumentowe co widać w tabeli. Popraw błędy. "

" [Enter] - wykres funkcji; [Inny dowolny klawisz] - wybór parametrów wykresu "

"Zapisz wyrażenie jeszcze raz ale w inny sposób abym wiedział co masz na myśli."

" [Enter] - wykres funkcji; [Inny dowolny klawisz] - wybór parametrów wykresu "

" Nie ma możliwości poprawiania oprócz usuwania ostatniego znaku [<--] "

Zdania dotyczące wyboru parametrów:

"Aktualnie wybierany parametr" <-- wskazuje go

"Naciśnij dowolny klawisz aby zobaczyć wykres funkcji"

"Naciśnij dowolny klawisz aby wybrać następny parametr"

"Podgląd [?] "

"Tak duża ilość znaków jest nie do przyjęcia"

"Popraw dane"

"Wielkość wprowadzona to nie liczba naturalna"

"Popraw dane"

"Liczba musi być z zakresu [10;300]"

"Popraw dane"

"Liczba musi być z zakresu [5;220]"

"Popraw dane"

"Liczba musi być z zakresu [0;11]"

"Popraw dane"

"Liczba musi być z zakresu [0;15]"

"Popraw dane"

"Kolor osi liczbowych jest ten sam co kolor tła"

"Popraw dane"

"Kolor wykresu funkcji jest ten sam co kolor tła"

"Popraw dane"

Zdania dotyczące wykresu funkcji:

"Ani jeden punkt wykresu nie pojawił się w widzianym obszarze."

"Pewien przedział wykresu funkcji wymaga płaszczyzny zespolonej." <-- to zdanie należy brać z 'przymrużeniem oka'.

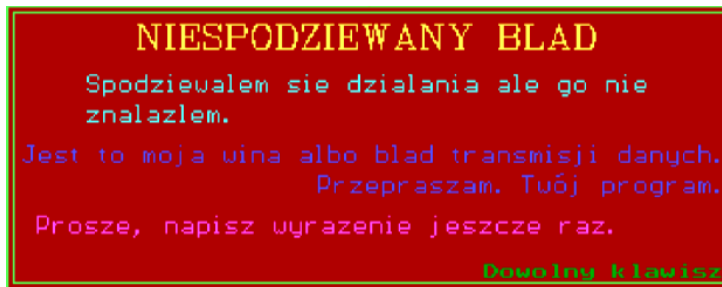
"Uwaga: '(' może okazać się być '[' a ')' ']''. Pytajnik oznacza że wykres zaczyna się lub kończy liczbą już należącą do dziedziny funkcji."

"Nie odkryłem punktów wykresu należących do dziedziny funkcji w zakresie osi rzędnych widzianych na ekranie monitora."

"W zakresie widzianym na ekranie nie wykryłem miejsc zerowych."

"W zakresie widzianym na ekranie nie wykryłem miejsc ekstremalnych."

W końcu, jeśli wszystko inne nie pozwoli na uzyskanie wykresu funkcji, pojawi się następujący komunikat:



Zdania dotyczące ostatecznego braku możliwości sporządzenia wykresu ("NIESPODZIEWANY BŁĄD"):

"Mam kłopot z nawiasami. Ich liczba lub pozycja nie zgadza się z wymaganiami obliczeń."

"Spodziewałem się funkcji ale jej nie znalazłem."

"Spodziewałem się działania ale go nie znalazłem."

"Spotkałem się z niezrozumiałym działaniem pomiędzy liczbami."

4. Zrozumieć grafikę

a) Rozdzielczość rozporządzana przez Turbo C++. W nawiasach numery pikseli.



b) Kolory - można deklarować je liczbą lub nazwą (wielkie litery)

0 - czarny BLACK	8 - ciemnoszary DARKGRAY
1 - niebieski BLUE	9 - jasnoniebieski LIGHTBLUE
2 - zielony GREEN	10 - jasnozielony LIGHTGREEN
3 - niebieskozielony/błękitny CYAN	11 - jasny cyjan LIGHTCYAN
4 - czerwony RED	12 - jasnoczerwony LIGHTRED
5 - purpurowy MAGENTA	13 - jasnopurpurowy LIGHTMAGENTA
6 - brązowy BROWN	14 - żółty YELLOW
7 - jasnoszary LIGHTGRAY	15 - biały WHITE

c) Rodzaje linii i nazwy grubości

Rodzaje linii	Grubość linii
-----	-----
SOLID_LINE	NORM_WIDTH

DOTTED_LINE	
CENTER_LINE	THICK_WIDTH
DASHED_LINE	
USERBIT_LINE	

d) Wzory rodzajów linii

SOLID_LINE	= 0
DOTTED_LINE	= 1
CENTER_LINE	= 2
DASHED_LINE	= 3

e) Nazwy wzorów

Nazwa	Powoduje wypełnienie przez
EMPTY_FILL	Kolor tła
SOLID_FILL	Jednolity kolor
LINE_FILL	Linie poziome -----
LTSLASH_FILL	Cienkie, ukośne położone linie /////
SLASH_FILL	Grube, ukośne położone linie /////
BKSLASH_FILL	Grube, ukośne położone odwrócone ukośniki \\\\\\\
LTBKSLASH_FILL	Cienkie ukośniki odwrotne \\\\\\\
HATCH_FILL	Cienkie kreskowanie krzyżowe
XHATCH_FILL	Grube kreskowanie krzyżowe
INTERLEAVE_FILL	Linie przeplatające się
WIDE_DOT_FILL	Szeroko rozstawione kropki
CLOSE_DOT_FILL	Gęsto rozmieszczone kropki

f) Style tekstów

Aby ustawić charakterystyczne wartości tekstu użyj komendę **settextstyle**(*czcionka*, *kierunek*, *wielkość liter*);

Czcionka	Kierunek	Rozmiar czcionki
DEFAULT_FONT	HORIZ_DIR <-- Z lewej do prawej	1 = Domyślny (podstawowy)
TRIPLEX_FONT	VERT_DIR <-- Z dołu do góry	2 = Rozmiar podwójny
SMALL_FONT		3 = Rozmiar potrójny
SANS_SERIF_FONT		4 = Rozmiar poczwórny
GOTHIC_FONT		5 = 5 razy podstawowy
SCRIPT_FONT		...
SIMPLEX_FONT		10 = 10 razy podstawowy
TRIPLEX_SCR_FONT		
COMPLEX_FONT		
EUROPEAN_FONT		
BOLD_FONT		

g) Justowanie tekstu

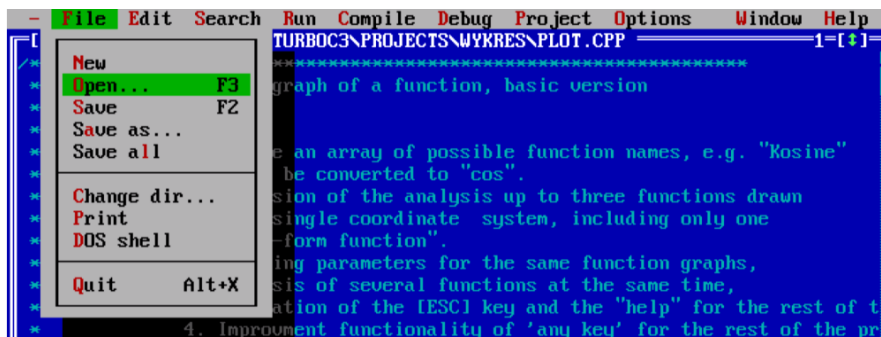
Aby podać w jaki sposób ma być tekst ustawiony wokół podanej pozycji użyj komendę **settextjustify**(*poziomo*, *pionowo*);

Poziomo	Pionowo
LEFT_TEXT	TOP_TEXT
CENTER_TEXT	BOTTOM_TEXT
RIGHT_TEXT	

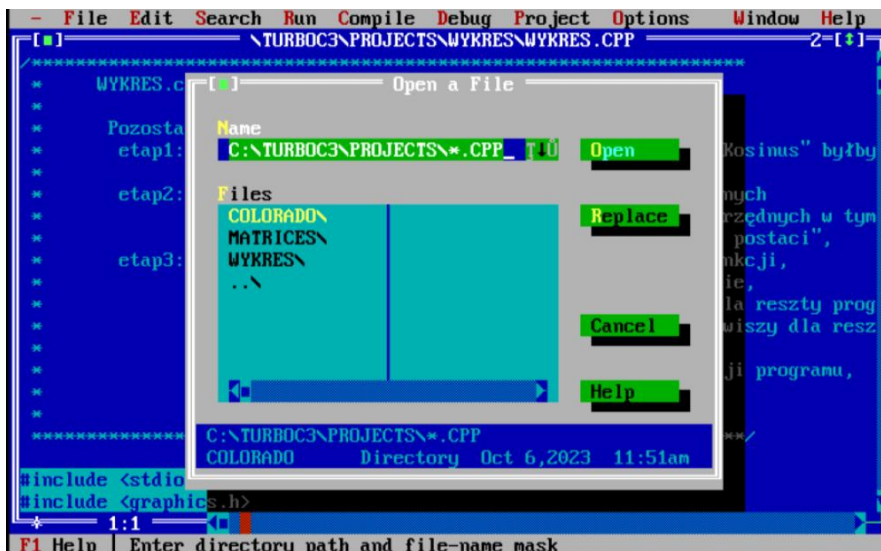
Trochę więcej o nawigacji w środowisku *Borland Turbo C++ v. 3.0* czyli inne, ważne elementy Menu.

Przejść do Menu możesz poprzez klawisz funkcyjny [F10] albo [Alt]+odpowiednia litera albo [Alt]+[F] i strzałką [→] na wybraną opcję.

File



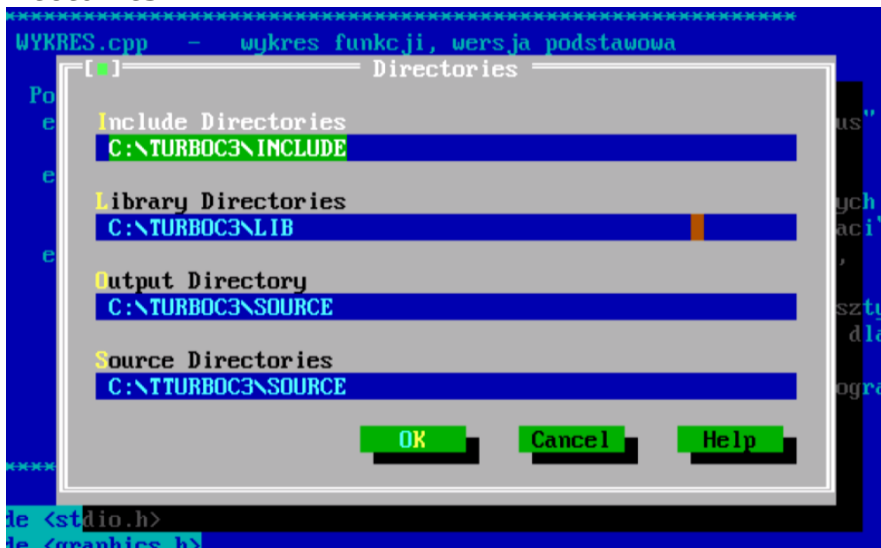
File --> Open



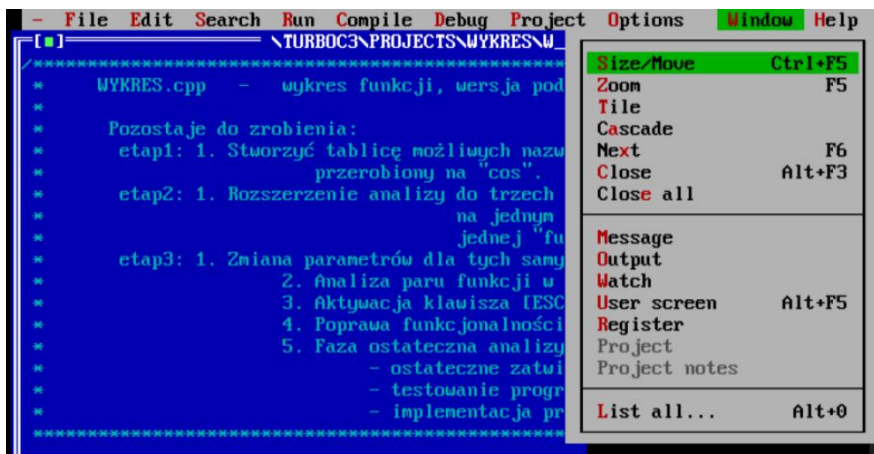
I używaj klawiszy [Tab] i [Enter].

Poniżej wyszczególnione są najważniejsze foldery dedykowane dla odpowiednich plików:

Options --> Directories

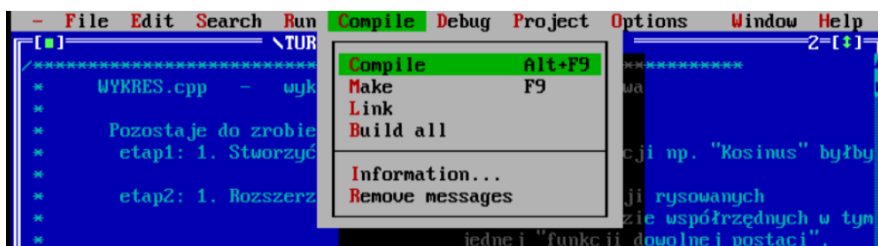


Można ukierunkować **Output** gdzie indziej, ale po co to robić?



Jak nie przytrzymasz wyświetlonego wyniku komendą **getch()** tuż na końcu programu, to mignie ten obraz (niemal niezauważalny) i z powrotem będziesz widzieć edytor. Wtedy **User screen** da podgląd na ten wynik.

Zoom zamyka **Watch**. **[F5]** powoduje także zamykanie **Watch** a następne **[F5]** jego otwieranie.

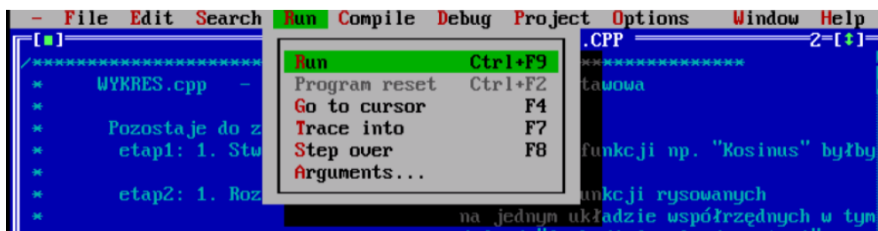


Make tworzy plik egzekucyjny. Oznacza to, że taki plik możesz od razu uruchomić (np. z panelu/desktopu), tak jak np. dowolny plik w **MSWord**. Możesz go przesłać na inny komputer i stamtąd go uruchomić - ale, w tym wypadku musisz mieć **C:\TURBOC3\bgi** tak, jak zadeklarowane jest w programie:

```
initgraph(&driver, &mode, "c:\\TURBOC3\\bgi"); // możliwość wymiany tego, co w cudzysłowie
```

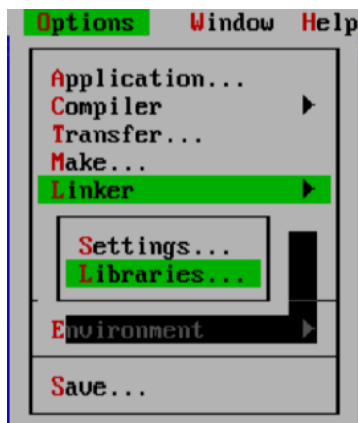
Niestety bez tej deklaracji **Wykres.cpp** mi nie działa. Może Tobie uda się to zrobić inaczej!

Ale to wszystko jest tylko teorią! Musiałbyś mieć stary komputer ze starym Windows'em!



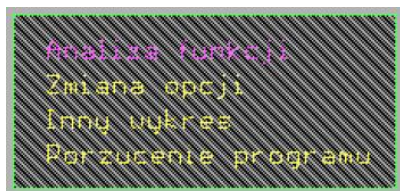
Najczęściej wchodzę do **Help** bo mam tam wszystko co chcę, łącznie z przykładami kodów.

Aby być pewnym, że Turbo C++ ma do programu podłączoną grafikę, kliknij na **Options** -> **Linker** -> **Libraries**. Zaznacz opcję **Graphics Library** i zaakceptuj to przyciskając **OK**.

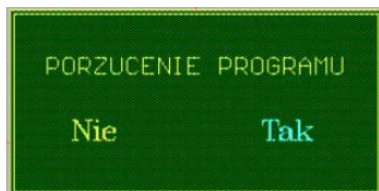


Niektóre mankamenty programu.

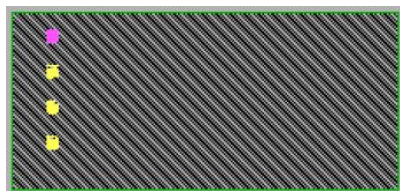
1. Myszka nie jest programowo podłączona a więc nie działa.
 2. Nie można wyjść daleko poza początek układu współrzędnych.
 3. Wykres to cienkie linie i wydawało by się, że powinno się je pogrubić. Technicznie jest to bardzo proste. Jednak powodowałoby to niekiedy niepoprawny wizualnie odczyt z tych linii. Np. $f(x)=\sin(x)/x$, tu powinno być widoczne, że $X=0$ nie należy do dziedziny funkcji. Co zrobić jeżeli wykres ślizga się po linii układu współrzędnych - w tym wypadku linie układu współrzędnych nie powinny być pokryte wykresem.
 4. Linia, której następny punkt wykracza poza górną lub dolną część ekranu a funkcja jest tu nadal ciągła, nie jest rysowana. Można by było ją narysować do granicy ekranu dla tej samej co poprzednio wartości X, lub $X + \text{jeden piksel}$. Podobnie można by narysować linię z punktu spoza ekranu na ekran nie powodując przekłamań programowych.
 5. Nie zawsze dobrze wpisana funkcja da wykres. Akurat ten mankament jest także w innych aplikacjach internetowych.
 6. Funkcja **outtext()** nie przyjmuje nieangielskich znaków, chociaż można by było z tym sobie poradzić.
 7. Dlaczego powtórne wywołanie modułu **menu()** nie daje pożądanej frazy **outtext**'u?
- Pierwsze wywołanie **menu()** na wykresie funkcji daje poprawny rezultat:



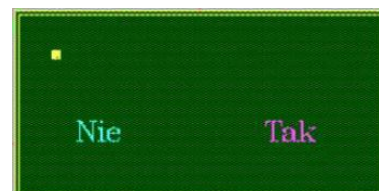
i



Po wejściu do '**Analiza funkcji**', sprawdzeniu wyników, wychodzimy ponownie do pełnego obrazu wykresu funkcji. Gdy jednak ponownie chcemy wejść do tego samego **menu()** bo np. zapomnieliśmy danych tam wypisanych, otrzymamy to:



i



menu() nadal będzie działać i punkt pierwszy jest nadal czynną '**Analizą funkcji**' dającą po **[Enter]** poprzednie dane.

Ale jak widać, to ponowne wywołanie **menu()** nie jest przyjemne dla oka. Trzeba pamiętać kolejność opcji:

1. Analiza funkcji
2. Zmiana opcji
3. Inny wykres
4. Porzucenie programu.

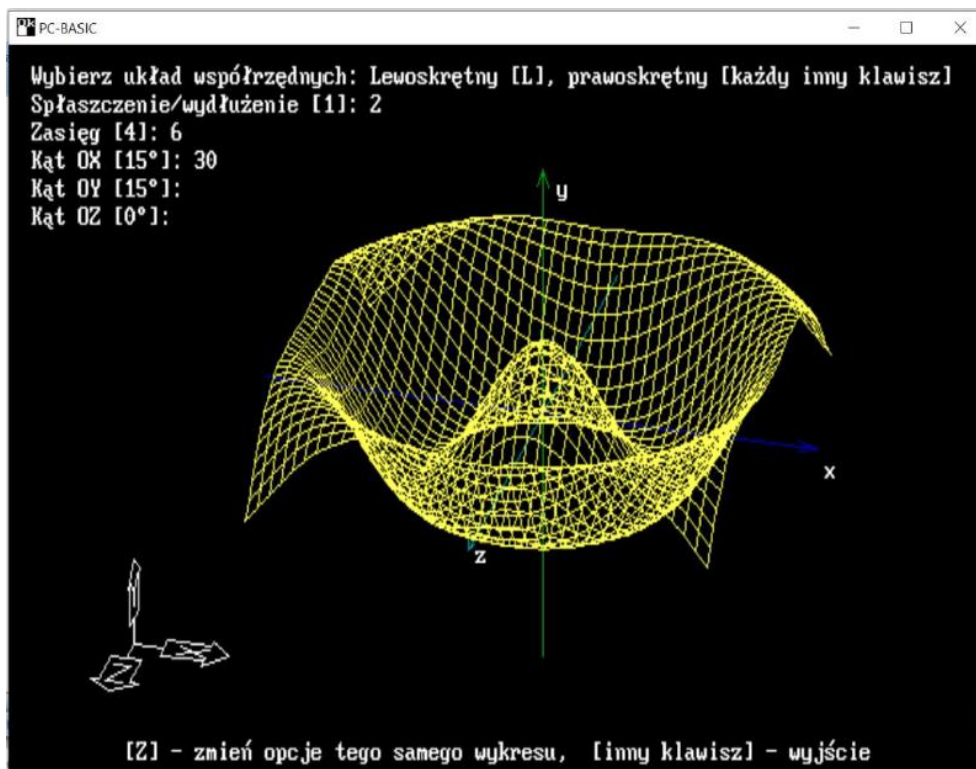
W programie nie ma logicznego błędu. Tłumaczenie znalazłem dopiero teraz w Internecie (w 1998 roku, gdy tworzyłem ten program, nie miałem dostępu do Internetu):

The **outtext()** function is a C function that displays text at the current position on the screen in graphics mode. However, **it only works once** for each element that you want to display. If you try to use it twice for the same element, it will not work because the element is already in the document and cannot be appended again. You need to create a new element for each text that you want to display, or use the **cloneNode()** method to make a copy of the existing element. Alternatively, you can use the **outtextxy()** function, which displays text at a specified point (x, y) on the screen. This function allows you to display multiple texts at different positions without creating new elements.

Moją intencją jest jednak przekazanie logiki programowania a nie kombinowanie 'jak koń pod górę'.

A co z grafiką trójwymiarową?

Dla wykresu figury przestrzennej jest program 'XYZ' w GW-Basic poniżej tej strony internetowej, oczywiście z pełnym kodem. To był mój pierwszy program na PC napisany ponad 30 lat temu (1991r.). I tu od razu przykład działania programu dla funkcji $\cos(\sqrt{x^2+y^2})$ <-- w GW-Basic zamiast 'sqrt' jest 'sqr'. Nie dziw się więc, że w programie **Wykres.cpp** zrobiłem sobie z tego 'sq' !



Porównanie nazw wersji polskiej z angielską

Program

Polski	Angielski
Wykres.cpp	Plot.cpp

Funkcje

Polski	Angielski
void main(void)	void main(void)
void wprowadzenie(void);	void introduction(void);
void wybieranie_postaci_funkcji(void);	void function_type_selection(void);
void okno_najwieksze(void);	void the_largest_window(void);
void wybieranie_wielomianu(void);	void polynomial_choosing(void);
void usun_spacje_i_analizuj(void);	void remove_spaces_and_analyze(void);
void popraw_zapis_liczby(void);	void correct_notation_of_the_number(void);
void zbadaj_wielkosc_liczby(void);	void check_number_size(void);
void wybieranie_funkcji_dowolnej(void);	void free_form_function_choosing(void);
void pytanie_o_zapis(void);	void question_about_notation(void);
void tablica_operatorow_i_funkcji(void);	void operators_and_math_func_table(void);
void grafika_wstępu(void);	void introduction_graphics(void);
void wybory_parametrow(void);	void parameters_selection(void);
void dlugosc_jednostki_osi_OX(void);	void OX_axis_unit_length(void);
void dlugosc_jednostki_osi_OY(void);	void OY_axis_unit_length(void);
void nr_wzoru_tla_ekranu(void);	void background_pattern_number(void);
void okno_wzorow_tla_i_barw(void);	void bgr_patterns_and_colors_window(void);
void nr_koloru_tla_ekranu(void);	void background_color_number(void);

void nr_koloru_osi_ukladu_wspolrz(void);	void coor_system_axes_color_number(void);
void nr_koloru_wykresu_funkcji(void);	void function_graph_color_number(void);
void wstep_rutynowy(void);	void routine_introduction(void);
void wstepna_analiza_znakow(void);	void preliminary_analysis_of_chars(void);
void brak_mozliwosci_poprawiania(void);	void no_possibility_of_correction(void);
void tlo_wykresu_funkcji(void);	void function_graph_background(void);
void wykres_funkcji(void);	void graph_of_the_function(void);
Raport błędów	
void nie_liczba_rzeczywista(void);	void this_is_not_a_real_number(void);
void za_dlugi_lancuch(void);	void string_is_too_long(void);
void usun_spacje(void);	void remove_spaces(void);
void nie_liczba_naturalna(void);	void this_is_not_a_natural_number(void);
void liczba_poza_zakresem(void);	void out_of_range_number(void);
void kolor_ten_sam(void);	void the_same_color_is_not_allowed(void);
void wyczysc_raport_bledu(void);	void clear_error_report(void);
void wyczysc_informacje(void);	void clear_info(void);
void znaleziono_blad(void);	void error_found(void);
Porzucenie programu i pomoc	
void czy_porzucic_program(void);	void whether_to_exit_the_program(void);
void zdania_sprzeczne_ze_soba(int ilosc_skokow);	void contradictory_sentences(int hop_count);
void czy_porzucic_program_gr(void);	void whether_to_exit_the_program_gr(void);
void pomoc(void);	void help_non_gr(void);
void pomoc_gr(void);	void help_gr(void);
Funkcje i ich typy dla "funkcji dowolnej"	
void wpisz_lancuch_i_analizuj_go(void);	void read_the_string_and_analyze_it(void);
void popraw_wyrazenie(void);	void correct_algebraic_expression(void);
void czy_jest_argument(void);	void is_there_any_argument(void);
void dzialania_w_nawiasie_wewn_test(void);	void inner_parenth_operations_test(void);
void dzialania_w_nawiasie_wewn(void);	void inner_parenth_operations(void);
void usun_tylko_nawiasy(void);	void remove_only_parentheses(void);
void wprowadz_liczbe_do_lancucha(void);	void insert_number_into_the_string(void);
float wynik_prostego_dzialania(float aa, float bb);	float result_of_a_simple_operation(float aa, float bb);
Inne podstawowe (oprócz wpisz_lancuch_i_analizuj_go) elementy "funkcji dowolnej postaci"	
void funkcja_dowolnej_postaci(void);	void free_form_function(void);
void oblicz_wartosc_funkcji_matem(void);	void math_function_calculation(void);
int czy_brak_obliczen(void);	int is_it_a_lack_of_calculations(void);
Okno "menu" po wykonanym wykresie	
void menu(void);	void function_menu(void);
void analiza_funkcji(void);	void function_analysis(void);
void okno_analazy_funkcji(void);	void window_of_function_analysis(void);

Zmienne

Polski	Angielski	Uwagi
int opcja = 0;	int option = 0;	Początkowo 0 - wartość bez znaczenia; Wartość operacyjna to od 1 do 7 do zarządzania wyborem parametrów.
int kolor_tla_ekranu_opcji = 11;	int backgr_color_for_option = 11;	Kolor tła ekranu do wyboru parametrów; początkowo jasnoniebieski.
int kolor_tla_wyboru_parametrow = 7;	int backgr_color_for_parameter_sel = 7;	Kolor tła ekranu do wyboru parametrów; początkowo jasnoszary.
int zle;	int bad;	Flaga: 0 - OK; 1 - coś jest źle. Wezwanie do korekty zapisu.
char znak;	char character_entered;	Pojedynczy znak wprowadzony z klawiatury dla funkcji getch().
char *ptr_znak;	char *ptr_character_entered;	Wskaźnik dla zmiennej 'znak'.
int dl_lancucha;	int string_length;	Długość ciągu znaków wprowadzona z klawiatury przez użytkownika.
int proba_poprawienia = 0;	int trying_to_correct = 0;	Flaga: 0 - OK; 1 - konieczna korekta.
int pozycja_kursora_x;	int cursor_position_x;	Pozioma pozycja kursora dla ekranu niegraficznego; wzięta z wbudowanej funkcji wherex().
Int pozycja_kursora_y;	Int cursor_position_y;	Pionowe położenie kursora dla ekranu niegraficznego; pobierane z wbudowanej funkcji wherey().

int sygnal = 0;	int beep_attention = 0;	Uniwersalna flaga do określonego zastosowania. Wezwanie do korekty zapisu.
Wybór parametrów funkcji		
int dl_jedn_OX = 50;	int unit_length_on_the_OX = 50;	Wartość operacyjna „Długość jednostkowa na osi OX w pikselach” dla końcowego wykresu; początkowo 50; dopuszczalny zakres [10;300].
int dl_jedn_na_osi_OX = 50;	int unit_length_on_the_OX_axis = 50;	To samo co powyżej, ale podczas rysowania funkcji. Tutaj: ustalona wartość bez zmian.
int dl_jedn_OY = 50;	int unit_length_on_the_OY = 50;	Wartość operacyjna „Długość jednostkowa na osi OY w pikselach” dla końcowego wykresu; początkowo 50; dopuszczalny zakres [5;220].
int dl_jedn_na_osi_OY = 50;	int unit_length_on_the_OY_axis = 50;	To samo co powyżej, ale podczas rysowania funkcji. Tutaj: ustalona wartość bez zmian.
int zmniejszenie_jednostki_OY = 1;	int unit_OY_length_reduction = 1;	1 - nie zmieniaj 'długości jednostki na osi OY w pikselach' wprowadzonej dla końcowego wykresu; Inne - zmniejsz „długość jednostki na osi OY w pikselach”, którą wpisałeś 10 razy.
int nr_wz_tla = 1;	int backgr_patt_number = 1;	Wartość operacyjna „Numeru wzoru tła” dla końcowego wykresu; 1 - jednolity kolor; dopuszczalny zakres wartości [0;11].
int nr_wzoru_tla = 1;	int backgr_pattern_number = 1;	To samo co powyżej, ale podczas rysowania funkcji. Tutaj: ustalona wartość bez zmian.
int nr_kol_tla = 7;	int backgr_col_number = 7;	Wartość operacyjna „Numeru koloru tła” dla końcowego wykresu; 7 - jasnoszary; dopuszczalny zakres wartości [0;15].
int nr_koloru_tla = 7;	int backgr_color_number = 7;	To samo co powyżej, ale podczas rysowania funkcji. Tutaj: ustalona wartość bez zmian.
int nr_kol_osi = 12;	int axes_col_number = 12;	Wartość operacyjna „Numeru koloru osi układu współrzędnych” dla końcowego wykresu; 12 - jasnoczerwony; dopuszczalny zakres [0;15].
int nr_koloru_osi = 12;	int axes_color_number = 12;	To samo co powyżej, ale podczas rysowania funkcji. Tutaj: ustalona wartość bez zmian.
int nr_kol_wykresu = 10;	int function_plot_col_number = 10;	Wartość operacyjna „Numeru koloru wykresu funkcji” dla wykresu końcowego; 10 - jasnozielony; dopuszczalny zakres [0;15].
int nr_koloru_wykresu = 10;	int function_plot_color_number = 10;	To samo co powyżej, ale podczas rysowania funkcji. Tutaj: ustalona wartość bez zmian.
Dla wyboru 'wielomianu'		
int ilosc_kropek_dziesietnych;	int number_of_decimal_points;	Tylko dla 'Wyboru wielomianu'.
int wielomian = 0;	int polynomial_chosen = 0;	Flaga: 0 - nie wybrano wielomianu, 1 - wybrano wielomian - przeciwieństwo 'wybieranie funkcji dowolnej'.
Dla wyboru 'funkcji dowolnej postaci'		
int funkcja_dowolna = 0;	int free_form_function_chosen = 0;	Flaga: 0 - nie wybrano funkcji dowolnej, 1 - wybrano funkcję o dowolnej formie - przeciwieństwo 'wielomian'.
int jest_cyfra = 0;	int it_is_a_digit = 0;	Flaga: 0 - to nie jest cyfra, 1 - wykryto cyfrę.
int jest_x = 0;	int x_found = 0;	Flaga: 0 - nie odnaleziono zmiennej (x), 1 - odnaleziono zmienną (x).
int zmniejsz_d_l;	int decrease_d_l;	Licznik: Ile miejsc we wprowadzonym łańcuchu znakowym powoduje zmniejszenie jego długości. Np.: poprzez usuwanie spacji.
int wskaznik;	int indicator;	Uniwersalny wskaźnik wprowadzonego ciągu - wykrywa 'x', cyfrę, ...
int dodawanie = 0;	int addition = 0;	Flaga: 0 - to nie jest dodawanie, 1 - jest to dodawanie.
int odejmowanie = 0;	int subtraction = 0;	Flaga: 0 - to nie jest odejmowanie, 1 - jest to odejmowanie.
int mnozenie = 0;	int multiplication = 0;	Flaga: 0 - to nie jest mnożenie, 1 - jest to mnożenie.
int dzielenie = 0;	int division = 0;	Flaga: 0 - to nie jest dzielenie, 1 - jest to dzielenie.
int dzialanie;	int is_there_any_operator_indic;	Flaga: 0 - to nie jest znak jeden z: +, -, * i / 1 - wykryto jeden z następujących znaków działania: +, -, * i /.
int nr_nawiasu_otwartego;	int number_of_bracket_open;	Kolejny numer nawiasu otwartego.
int nr_nawiasu_zamknietego;	int number_of_bracket_closed;	Kolejny numer nawiasu zamkniętego.
int poczatek_operacji;	int operation_begins;	Pozycja we wpisanym łańcuchu, od której rozpoczynają się obliczenia, np. nawiasy najbardziej wewnętrzne.
int koniec_operacji;	int operation_ends;	Pozycja we wpisanym łańcuchu, na której kończą się obliczenia, np. nawiasy najbardziej wewnętrzne.
int pozycja_nawiasu;	int parenthesis_position;	Pozycja w ciągu, w którym wstawiony jest jeden z nawiasów „(” lub „)”.

int jest_funkcja;	int it_is_a_function;	Detektor funkcji matematycznych. Flaga: 0 - na razie nie ma funkcji, 1 - tutaj zaczyna się funkcja matematyczna.
int jest_?;	int it_is_?;	gdzie: '?' może być jednym z: ln, lg, cos, sin, tan, cosh, sinh, tanh, acos, asin, atan, abs, mod, pow, exp, sq.
int poz_?_1;	int pos_?_1;	
int poz_?_2;	int pos_?_2;	
int operacje_na_funkcji = 0;	int operations_on_function = 0;	void oblicz_wartosc_funkcji_matem(void) Flaga: 0 - nie traktuj tego jako funkcji matematycznej, 1 - zacznij traktować to jako funkcję matematyczną i na koniec uruchom moduł oblicz_wartosc_funkcji_matem()
int ilosc_przecinkow_faktyczna;	int actual_number_of_commas;	Liczba przecinków wewnątrz funkcji mod() lub pow() w wprowadzonym łańcuchu znakowym.
int ilosc_przecinkow_przewidywana;	int predicted_number_of_commas;	Oczekiwana liczba przecinków w funkcji mod() lub pow() (=1).
Int jest_przecinek = 0;	int it_is_comma = 0;	Wskaźnik „przecinków”. Flaga: 0 - bez przecinka, 1 - to jest przecinek, 2 - gotowy do przyjęcia drugiego argumentu funkcji mod() lub pow().
int porzuc;	int terminate;	Flaga: 0 - nadal rysuj funkcję 1 - przestań rysować linię ze względu na konkretny problem, np. następny punkt jest poza ekranem lub następny punkt nie należy do dziedziny funkcji.
int jest_problem_z_dziedzina = 0;	int problem_with_domain = 0;	Flaga: 0 - nie ma problemu z dziedziną, 1 - problem z dziedziną, np. Liczba wyników potęgowania jest zbyt wysoka (to ograniczenie programu).
int blad = 0;	int error_detector = 0;	Uniwersalna flaga do wykrywania błędów. Jeśli stwierdzono błąd, wywoływana jest funkcja 'znaleziono_blad()'.
int dziedzina;	int function_domain;	Flaga: 0 - punkt 'x' nie należy do dziedziny funkcji, 1 - ten punkt 'x' należy do dziedziny funkcji.
int kwadrat_maksymalny[];	int maximum_square[];	void wprowadzenie(void) void grafika_wstepu(void) void tlo_wykresu_funkcji(void) Zestaw zmiennych uruchamiających ekran graficzny.
unsigned rozmiar;	unsigned image_size;	
void far *ptr_do_alokacji_pamieci;	void far *ptr_to_memory_allocation;	void usun_spacje_i_analizuj(void) Flaga: 0 - to nie jest cyfra, 1 - to jest cyfra.
int jest_cyfra = 0;	int it_is_a_digit = 0;	void popraw_zapis_liczby(void) Unikaj drukowania mało znaczących cyfr i usuwaj nadmiar kropek.
int jest_cyfra_znacząca = 0;	int it_is_a_significant_digit = 0;	void wpisz_lancuch_i_analizuj_go(void) Pobiera 'Długość wprowadzonego łańcucha znaków' do specjalnych celów, np. wpisując 3,141593 zamiast 'pi' (π) lub 2,718282 zamiast 'e'.
int usun_tylko_kropke = 0;	int delete_only_the_dot = 0;	
int d_l_fikcja;	int d_l_fiction;	
'Pomoc' - zarówno dla obrazu graficznego jak i niegraficznego		
int wezwanie_pomocy = 0;	int calling_for_help = 0;	Flaga: 0 - nie jest potrzebne okno pomocy, 1 - potrzebne okno pomocy. Zarówno dla ekranu graficznego, jak i
int strona;	int page_number;	Numer strony dla funkcji 'pomoc()'.
char another_character;	char znak_dodatkowy;	void pomoc(void) Dla [PgDn] i [PgUp] przy użyciu getch() w funkcji 'pomoc()'.
petla_?;	page_no_?;	void pomoc(void) etykiety (labels) Gdzie ? znaczy: 1, 2, 3, 4, 5 lub 6 Aby zarządzać sekwencją sześciu okien o tym samym rozmiarze dla funkcji 'pomoc()'.
rozdzielnia;	switchboard;	void pomoc(void) etykieta (label) Aby wspomagać pracę zestawu 'petla_?'.
int pole_pomocy_gr[];	int help_field_gr_box[];	void pomoc_gr(void) Definicja tworzenia okna dla tej funkcji.
Porzucenie programu		
int proba_porzucenia = 0;	int trying_to_exit = 0;	Flaga: 0 - gotowy, aby nie wychodzić z Programu, 1 - gotowy do wyjścia z programu. Zarówno dla ekranu graficznego, jak i niegraficznego.
int pole_porzucenia_programu[];	int exit_the_program_box[];	void czy_porzucic_program_gr(void) Definicja tworzenia okna dla tej funkcji.

int ilosc_skokow = 0;	int hop_count = 0;	Licznik przeskoków pomiędzy zdaniem dla funkcji 'zdania_sprzeczne_ze_soba(ilosc_skokow)'.
void wykres_funkcji(void)		
float granica_rozsadku;	float limit_of_reason;	Arbitralnie (zgodnie z moją własną logiką) narzucone górne (i automatycznie dolne jako 'minus górne') ilościowe ograniczenia na szereg zmiennych o wartościach w liczbach naturalnych i w dodatku inne ograniczenia liczbowe dla tej samej zmiennej ale odmiennie traktowanej. Np. 'długość jednostki osi OX' < 20 jest nią 1e+18 a dla > 20 i większych 'granice rozsądku' są liczby coraz mniejsze.
int yy_poprzedni;	int yy_previous;	Zachowana pozycja na ekranie punktu poprzedniego, aby się ewentualnie do niego odnieść. Wtedy zmienna jest_punkt = 1 i ewentualnie rysuje linię. Poza ekranem: Nie rysuje linii.
int punkt_poprzedni;	int point_previous;	Flaga: 0 - nie ma takiego punktu, 1 - jest punkt na ekranie. Czy w ogóle jest jakiś punkt poprzedni do którego można się odwołać? Np. gdy rysowanie jest przerwane - nieciągłość, wyjście poza ekran, itd - nie ma takiego punktu.
int punkt_poprzedni_poprzedni;	int point_before_previous;	Flaga: 0 - nie ma takiego punktu 1 - jest punkt na ekranie Czy w ogóle jest jakiś punkt przed punktem poprzednim do którego można się odwołać?
float y_poprzedni;	float y_previous;	Rzeczywista wartość 'Y' poprzedniego punktu.
float y_poprzedni_poprzedni;	float y_before_previous;	Rzeczywista wartość 'Y' punktu przed poprzednim.
float skladnik;	float summand;	Liczbowy składnik sumy.
float wartosc_piksela_OX;	float OX_pixel_value	Znajdź współrzędną X piksela na ekranie.
float dl_jedn_na_osi_OY_float;	float unit_length_on_the_OY_axis_flo;	Znajdź współrzędną Y piksela na ekranie. Wprowadzono aby uniknąć zaokrąglenia. (Znalezienie konkretnego piksela może spowodować, że linia nie będzie płynna.)
int jest_punkt = 0;	int point_flag;	Obliczono pozycję punktu i ... Flaga: 0 - ... punkt jest poza ekranem, 1 - ... punkt jest na ekranie. Dla pokazania, że skoncentrowano punkt na ekranie.
Niektóre inne zmienne		
int dziedzina_poprzednia;	int previous_function_domain;	Pomocnicza flaga do szacowania granic dziedziny funkcji dowolnej postaci.
int brak_obliczen = 0;	int lack_of_calculations = 0;	int czy_brak_obliczen(void) Flaga: 0 - nie zatrzymuj się i wykonaj Obliczenia, 1 - nie wykonuj obliczeń. Nie obliczaj, jeśli spełniony jest jakiś warunek.
int pole_menu[];	int menu_box[];	void menu(void) Definicja tworzenia okna dla tej funkcji.
int pole_analazy_funkcji[];	int function_analysis_box[];	void okno_analazy_funkcji(void) Definicja tworzenia okna dla tej funkcji.
int pole_bledu[];	int error_box[];	void znaleziono_blad(void) Definicja tworzenia okna dla tej funkcji.

Wpisywanie liczb z przecinkiem dziesiętnym

Wprowadzanie liczb z przecinkiem dziesiętnym zamiast kropki dziesiętnej może być łatwe: Nie zezwalaj na wpisywanie "kropki" i wprowadź wiersz kodu, który zastępuje wizualny "przecinek" "kropką", aby umożliwić obliczenia na wprowadzonej liczbie. Tak więc program ten jest dla nieangielskich użytkowników.

Jednocześnie każdy znak jest poddany weryfikacji.

Co prawda, liczbę można zweryfikować po jej wpisaniu ale kontrola nad każdym znakiem pozwala na lepszy przebieg tego procesu.

Ten program można użyć jako funkcję wywoływaną za każdym razem, gdy trzeba wpisać jakąś liczbę (może być to zarówno liczba całkowita jak i zmiennoprzecinkowa) z klawiatury.

Gdybyśmy potrzebowali wydrukować końcową (po obliczeniach w programie) liczbę z **przecinkiem dziesiętnym** a nie z kropką dziesiętną trzeba by było:

- zamienić liczbę na ciąg znaków (w 'C' jest to funkcja `itoa()` czyli integer **to** alphabetic),
- znaleźć pozycję kropki (jeżeli taka jest),
- zamienić kropkę na przecinek,
- wydrukować ten ciąg znaków (udając, że to liczba).

Przykładowy kod:

Turbo C++

Nie potrzebujemy tu używać funkcji `'setlocale()'` ale obecność jej nie stanowi tu problemu. Tym bardziej, że program ten jest przygotowany także dla środowiska Dev C++ po wymianie `clrscr();` na `system("cls");`

Skopiowanie tego programu do środowiska Turbo C++ będzie wymagało poprawę zapisu polskich znaków!

```

/*****
 * Plik: Liczba.C
 *****/
 * Celem programu jest wprowadzenie liczby zmiennoprzecinkowej do dalszych na
 * niej obliczeń.
 * Wprowadzenie liczby z klawiatury z jednoczesnym sprawdzeniem każdego jej znaku.
 * Ten program jest dla nieangielskich użytkowników ponieważ w liczbie ma występować
 * przecinek dziesiętny (np. 12,34) a nie kropka dziesiętna (np. 12.34).
 */

#include<stdio.h>
#include<conio.h>          // dla getch()
#include<ctype.h>          // dla isdigit()
#include<locale.h>         // dla znaków narodowych innych niż angielskie
#include<stdlib.h>         // dla funkcji system("cls") w Dev C++

float a;                  // ostatecznie wprowadzona liczba
char A[15];               // składanie cyfr aby zamienić ich ciąg na liczbę
int flag = 0;             // flag=0 <-- nie ma błędu; flag=1 <-- wystąpił błąd
int i=0, j, k, l;         // 'i' wprowadzono tylko po to aby wprowadzona pierwsza
                          // (i=0) liczba nie mogła być równa zeru, np. w wyrażeniu
                          // a*x**2 + b*x (dla równania kwadratowego)
                          // 'j' - kolejna cyfra w liczbie
                          // 'k' - zmienna potrzebna do wyczyszczenia informacji o błędzie
                          // 'l' - licznik przecinków w liczbie

void main()
{
    setlocale(LC_CTYPE, "Polish");
    // setlocale( LC_ALL, "German");    // Przykład ustawienia języka niemieckiego

    clrscr();                    // Wyczyść ekran
    // system("cls");              // dla 'Dev C++'

    printf("\n   Wprowadzanie liczby z automatycznym sprawdzeniem każdego znaku");
    printf("\n   W liczbie może być przecinek dziesiętny a nie kropka");
    printf("\n   Klawisz [ENTER] przyjmie liczbę. Pisz po polsku! \n");
    printf("\n   Po słowie POPRAW naciśnij dowolny klawisz w celu korekcy \n\n");
    printf("\n   Tu wpisz liczbę: ");

    j = 0;                      // kolejny element tablicy
    l = 0;                      // ilość przecinków
    while( ( A[j] = getch() ) != '\x0D') // wpisuj znaki a jak skończysz, przyciśnij
    // [Enter]

    { flag = 0;
    // if(i == 0)                // tylko, jeżeli liczba nie może być zerem
    // if(j == 0 && A[j]=='0')
    // { printf("   'a' ma być różne od 0   POPRAW");
    //   flag = 1;
    // }

    if(j == 0)                  // dla wszystkich liczb/współczynników
    { if( !(A[j] == '-' || A[j] == ',' || isdigit(A[j])) )
      { printf("   Dopuszczalne są - , cyfra   POPRAW");

```

```

        flag = 1;
    }
    else
        if( A[j] == ',' )           // zamień przecinek na kropkę do dalszych obliczeń
                                    //   na liczbie
        { A[j] = '.';
          l++;
        }
    }
    else
        if(A[j] == '.')
        { printf("          Pisz po polsku !          POPRAW");
          flag = 1;
        }
    else
        if( A[j] == ',' )           // zamień przecinek na kropkę do dalszych obliczeń
                                    //   na liczbie
        { A[j] = '.';
          l++;
          if( l > 1 )
          { printf("      Za dużo przecinków w liczbie  POPRAW");
            flag = 1;
          }
        }
    else
        if( !isdigit(A[j]) )
        { printf("          Tu ma być cyfra !          POPRAW");
          flag = 1;
        }
    }

if( flag == 1 )
{ getch();                                // przytrzymaj ekran na informacji o błędzie

  for(k=0;k<40;k++) printf("\b"); // cofnij kursor o 39 pozycji
  for(k=0;k<40;k++) printf(" ");  // wpisz 39 spacji wymazując informację
  for(k=0;k<40;k++) printf("\b"); // cofnij kursor o 39 pozycji
}
else
    j++;                                // dobry znak, przygotowanie do wpisania następnego znaku
                                        //   (zacznij nową pętlę 'while')
}
                                        // koniec pętli 'while'

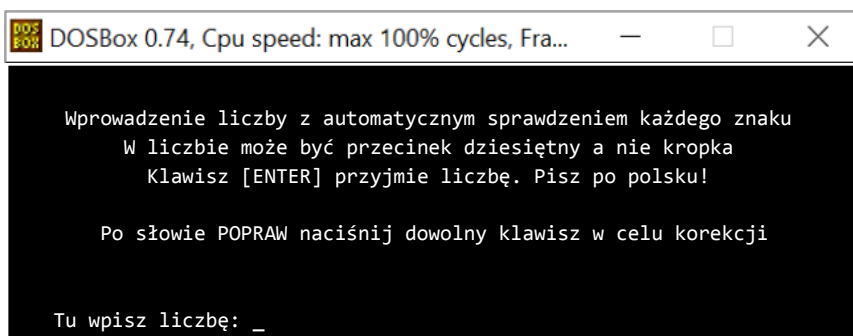
A[j++] = '\0';                          // zamknij łańcuch znakowy
a = atof(A);                             // wpisz łańcuch znakowy (A) do liczby (a)
printf("\n\n Program zaakceptował liczbę %f do dalszych na niej obliczeń\n\n",
a);

printf("\n\n          Naciśnij dowolny klawisz aby wyjść z programu ");
getch();

}
                                        // koniec funkcji 'main'

```

Możliwe wyniki:



DOSBox 0.74, Cpu speed: max 100% cycles, Fra...

```

Wprowadzenie liczby z automatycznym sprawdzeniem każdego znaku
W liczbie może być przecinek dziesiętny a nie kropka
Klawisz [ENTER] przyjmie liczbę. Pisz po polsku!

Po słowie POPRAW naciśnij dowolny klawisz w celu korekcji

Tu wpisz liczbę: _

```



```
DOSBox 0.74, Cpu speed: max 100% cycles, Fra...

Wprowadzenie liczby z automatycznym sprawdzeniem każdego znaku
W liczbie może być przecinek dziesiętny a nie kropka
Klawisz [ENTER] przyjmie liczbę. Pisz po polsku!

Po słowie POPRAW naciśnij dowolny klawisz w celu korekcji

Tu wpisz liczbę: +   Dopuszczalne są   - , cyfra   POPRAW
```

```
DOSBox 0.74, Cpu speed: max 100% cycles, Fra...

Wprowadzenie liczby z automatycznym sprawdzeniem każdego znaku
W liczbie może być przecinek dziesiętny a nie kropka
Klawisz [ENTER] przyjmie liczbę. Pisz po polsku!

Po słowie POPRAW naciśnij dowolny klawisz w celu korekcji

Tu wpisz liczbę: -,23456

Program zaakceptował liczbę -0.234560 do dalszych na niej obliczeń

Naciśnij dowolny klawisz aby wyjść z programu
```

Inny przykład:

```
DOSBox 0.74, Cpu speed: max 100% cycles, Fra...

Wprowadzenie liczby z automatycznym sprawdzeniem każdego znaku
W liczbie może być przecinek dziesiętny a nie kropka
Klawisz [ENTER] przyjmie liczbę. Pisz po polsku!

Po słowie POPRAW naciśnij dowolny klawisz w celu korekcji

Tu wpisz liczbę: 12.   Pisz po polsku !   POPRAW
```

```
DOSBox 0.74, Cpu speed: max 100% cycles, Fra...

Wprowadzenie liczby z automatycznym sprawdzeniem każdego znaku
W liczbie może być przecinek dziesiętny a nie kropka
Klawisz [ENTER] przyjmie liczbę. Pisz po polsku!

Po słowie POPRAW naciśnij dowolny klawisz w celu korekcji

Tu wpisz liczbę: 12,3m   Tu ma być cyfra !   POPRAW
```

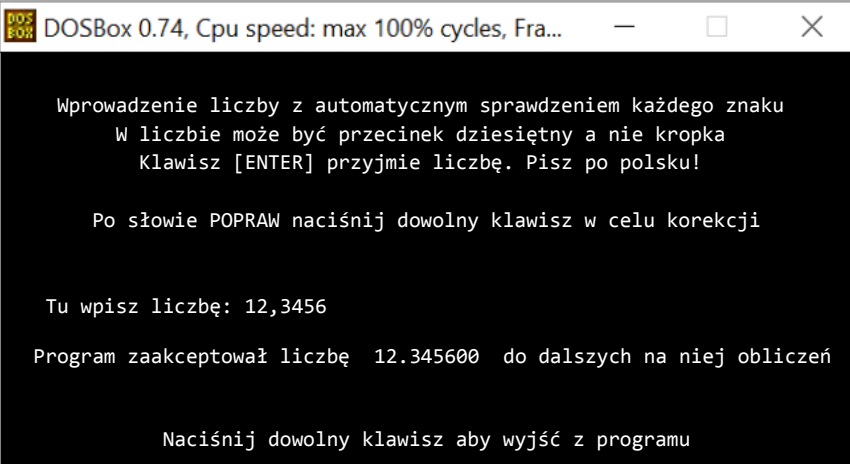
```
DOSBox 0.74, Cpu speed: max 100% cycles, Fra...

Wprowadzenie liczby z automatycznym sprawdzeniem każdego znaku
W liczbie może być przecinek dziesiętny a nie kropka
Klawisz [ENTER] przyjmie liczbę. Pisz po polsku!

Po słowie POPRAW naciśnij dowolny klawisz w celu korekcji

Tu wpisz liczbę: 12,3,   Za dużo przecinków w liczbie   POPRAW
```

W końcu:



DOSBox 0.74, Cpu speed: max 100% cycles, Fra...

Wprowadzenie liczby z automatycznym sprawdzeniem każdego znaku
W liczbie może być przecinek dziesiętny a nie kropka
Klawisz [ENTER] przyjmie liczbę. Pisz po polsku!

Po słowie POPRAW naciśnij dowolny klawisz w celu korekcji

Tu wpisz liczbę: 12,3456

Program zaakceptował liczbę 12.345600 do dalszych na niej obliczeń

Naciśnij dowolny klawisz aby wyjść z programu