QBASIC – procedury i funkcje

Funkcje operujące na znakach - wybrane

CHR$ i ASC

Są to funkcje pozwalające na zamianę kodów ASII
na odpowiadające im znaki – CHR$
oraz na znajdowanie kodów ASCII podanych znaków - ASC.

*CHR**$(Numer_kodu)***

Zwraca literę (łańcuch jednoznakowy) odpowiadającą danemu numerowi kodu ASCII.
Argumentem funkcji może być dowolny kod ASCII, liczba od 0 do 255. Wynikiem jest znak odpowiadający kodowi (liczbie).

Np. print chr$(77) **da w wyniku** M

*ASC**(wyrazenie_lancuchowe)***

Zwraca nr kodu ASCII pierwszej litery w wyrażeniu łańcuchowym

Przykłady programów z ASC i CHR$

Przykład 1:

```
start:    ' lub [start]
iNPUT "znak"; zn$  ‘ Wprowadzamy jakiś znak z klawiatury
PRINT ASC(zn4)                    ‘ wydruk kodu znaku (liczby)
GOTO start
```

Przykład 2

```
' Wydruk wszystkich znaków ASCII
CLS
PRINT "Kody ASCII i znaki "
FOR i = 0 TO 255
PRINT "kod: ", i, "  znak ", CHR$(i)   ‘ Wyświetla znak (np. literę) dla kolejnych cyfr od 0 do 255
NEXT i
```

Inne funkcje operujące na znakach:

*LEFT**$(wyrazenie_lancuchowe, n)***
- Zwraca n pierwszych znaków wyrażenia łańcuchowego
*RIFHT**$(wyrazenie_lancuchowe, n)*** ***- Zwraca n ostatnich znaków wyrażenia łańcuchowego***

np. a$="dzien dobry" : print right$(as$, 5)
‘ Na ekranie będzie dobry

*MID$(**wyrazenie_lancuchowe, start, [dlugosc]**)*
- Funkcja zwraca łańcuch wycięty z wnętrza łańcucha

np. print mid$("Ala ma kota", 1, 3) **- Wynik: Ala**

*LEN**(wyrazenie_lancuchowe)***
‘Zwraca ilość znaków w wyrażeniu łańcuchowym,

Podprogram (procedura) a funkcja

*Podprogram SUB **(inaczej** procedura**) - może wykonać wiele operacji i nie zwraca konkretnej jednej wartości,***
*Funkcja FUNCTION **zwraca jedną konkretną wartość, choć może też wykonać wiele operacji***

Składnia podprogramu **(procedury)**

*SUB **nazwa_ogolna (lista_parametrow_formalnych)***
...
END SUB
Np.

```
SUB drukuj (x)
        PRINT "Jestem w podprogramie DRUKUJ ";
        PRINT "x="; x, "a="; a, "b="; b
END SUB
```

Deklaracja procedury na początku programu

DECLARE SUB nazwa_ogolna (lista_parametrów_formalnych)

Np.

```
DECLARE SUB drukuj (x!)
```

' ! oznacza single precision – real,
‘ x! – parametr formalny

Wywołanie podprogramu

CALL nazwa_podprogramu (parametry_aktualne)
Np. CALL drukuj(b)
‘ Wywołanie podprogramu z parametrem aktualnym b

Przekazywanie wartości zmiennych między podprogramami

Podprogram musi być wywołany z odpowiednimi argumentami.

np. instrukcja wywołania CALL podpr1(a, b)
powoduje przejście do wykonywania podprogramu SUB podpr1(x,y)
Wartości zmiennych a, b (argumenty) w chwili przejścia do instrukcji wywołania, są przekazane do zmiennych x i y w podprogramie.
Zmiana wartości x, y, w wyniku obliczeń wykonywanych w podprogramie, nie powoduje jednoczesnych zmian wartości
zmiennych a i b w programie wywołującym.
Dopiero powrót do programu wywołującego powoduje, ze zmienne a, b przyjmują wartości, jakie w momencie opuszczania podprogramu
miały w nim zmienne x i y.
Zmienne o tych samych nazwach występujące w programie i podprogramach, jeśli nie są umieszczone jako argumenty w instrukcji wywołania,
są zupełnie od siebie niezależne,
chyba, że umieścimy w podprogramie specjalne deklaracje SHARED**.**

Reguły te można sprawdzić z przebiegu wykonania poniższego programu

Przykład z podprogramem drukuj

```
REM abcq13.bas  - podprogram drukuj – 2 wywołania
DECLARE SUB drukuj (x!)   ' ! oznacza single precision - real
CLS
a = 7
b = 15
CALL drukuj(a)
PRINT "Pierwszy powrót do Main"
CALL drukuj(b)
PRINT "Drugi powrót do Main"
PRINT "w programie MAIN a="; a, "b="; b, "x="; x
END

SUB drukuj (x)
PRINT "Jestem w podprogramie DRUKUJ ";
PRINT "x="; x, "a="; a, "b="; b
END SUB
```

Wyniki uruchomienia programu

```
Jestem w podprogramie DRUKUJ x=7        a=0         b=0
Pierwszy powrót do Main
Jestem w podprogramie DRUKUJ x=15       a=0         b=0
Drugi powrót do Main
w programie MAIN a=7       b=15        x=0
```

Deklaracja COMMON SHARED, SHARED

Składnia;
COMMON [SHARED][/blok/] zmienna [({rozmiar})] [AS typ] [, ...]
***Opis:** Definiuje zmienne globalne wspólne dla różnych modułów tego samego programu lub różnych programów powiązanych ze sobą.*
Parametry:
SHARED - ***określa, że zmienne tego modułu będą dostępne w całym programie***
blok - ***nazwa identyfikująca grupę zmiennych; maks. 40 znaków***
zmienna - ***nazwy zmiennych***
rozmiar - ***dowolna stała podająca liczbę wymiarów dla zmiennych tablicowych***
AS typ - ***deklaruje zmienne jako typu:*** *INTERGER, LONG, SINGLE, DOUBLE, STRING* ***lub wg własnej definicji.***
Przykład programu ze zmiennymi SHARED

```
REM ABCQ14.BAS
REM 2 podprogramy: druk1 i druk2
```

```
DECLARE SUB druk1 ()
DECLARE SUB druk2 ()
COMMON SHARED ws
CLS
ws = 23
PRINT "W programie glownym ws="; ws
CALL druk1    ' wywołanie podprogramu druk1 bez podawania argumentów
CALL druk2    ' wywołanie podprogramu druk2 bez podawania argumentów
END
SUB druk1
PRINT "w SUBroutine druk1 ws="; ws
END SUB
SUB druk2
PRINT "Tu takze (w subroutine druk2)  ws="; ws
END SUB
```

Funkcja

Funkcja ***jest*** ***oddzielnym modułem programu, w którym mogą być wykonywane różne operacje, a ich wynik przekazywany do programu głównego poprzez nazwę funkcji.***

Deklaracja funkcji:

DECLARE FUNCTION nazwafunkcji ***(parametry)***

Wywołanie funkcji:

Nazwafunkcji ***(parametry_aktualne)***

Przykład programu z funkcją **sumprz(a, b, c)**

Przykład: obliczenie sumy długości przekątnych prostopadłościanu o danych krawędziach, z wykorzystaniem funkcji.

```
REM Zastosowanie funkcji – FUNCTION nazwa (parametry)
DECLARE FUNCTION sumprz (a, b, c)
CLS
INPUT "Podaj 3 liczby "; a, b, c
PRINT "Suma dlug. przekątnych  prostopadloscianu  a,b,c= "; sumprz(a, b, c)
END  ‘ Koniec programu głównego

FUNCTION sumprz (a, b, c)
p1 = SQR(a ^ 2 + b ^ 2)
p2 = SQR(b ^ 2 + c ^ 2)
p3 = SQR(a ^ 2 + c ^ 2)
sumprz = 2 * (p1 + p2 + p3)
END FUNCTION
```