Стислий екскурс в історію алгоритмічної мови С

1972 р. - співробітник фірми Bell Laboratories Деніс Рітчі створив алгоритмічну мову С.

1973 р. - на мові С Деніс Рітчі реалізував операційну систему Unix

Середина 80-х рр. - Б'ярн Страуструп розробив мову «С з класами», що надалі стали називати мовою С++

Мова С++ є мовою високого рівня і основою багатьох систем програмування: Borland C++, Visual C++, Borland C++ Builder.

Елементи мови Сі

Елементи будь-якої мови:

- символи (алфавіт мови) - це основні неподільні знаки, за допомогою яких пишуться всі тексти на мові програмування;
- слова (лексеми) - мінімальні одиниці мови, які мають самостійний зміст;
- словосполучення (вирази) задають правило обчислення деякого значення;
- речення (оператори) задають кінцевий опис деякої дії.

Алфавіт

- великі та малі латинські літери: A-Z, a-z.

  Компілятор мови Сі розглядає літери верхнього та нижнього регістрів як різні символи;

- арабські цифри;

- символи: графічні та ескейп-послідовності (символи табуляції, символ переходу на наступний рядок тощо);

- символи , . ; : ? ' ! | / \ ~ ( ) [ ] { } < > # % ^ & - + * =

Лексеми – ідентифікатори, ключові слова, константи, рядки,

знаки операцій.

Ідентифікатори

Ідентифікатори використовуються для іменування різних об'єктів:

- змінних,
- констант,
- міток,
- функцій тощо.

При записі ідентифікаторів можуть використовуватися

- великі та малі латинські літери;
- арабські цифри;
- символ підкреслення.

Приклад: *Sum sum sUm SUM sUM*

Наочність та зрозумілість *file_name*

Константи

Константами називають сталі величини, які в процесі виконання програми не змінюються.

- цілі
  - десяткові (Приклад: 3, 10, 123, 1024);
  - вісімкові (починаються з 0, після якого розміщуються цифри 0-7, Приклад: 023, 0701);
  - шістнадцяткові (починаються з 0x або 0X, після яких розміщуються цифри 0-F, Приклад: 0xF123, 0X10A);
- дійсні

[ціла_частина][. дробова_частина][E [-] степінь]

Приклад: 2.2 , 220e-2, 22.E-1, .22E1

– символьні - це один або декілька символів, які заключені в апострофи. Приклад: ‘A’ ‘*’ ‘\n’

Послідовності символів, які починаються з символу \ (обернений слеш) називаються керуючими або escape-послідовностями

Спеціальний символ	Шістнадцятковий код	Значення
\a	07	звуковий сигнал
\b	08	повернення на 1 символ
\f	0C	переведення сторінки
\n	0A	перехід на наступний ряд
\r	0D	повернення каретки
\t	09	горизонтальна табуляція
\v	0B	вертикальна табуляція
\\	5C	символ \
\'	27	символ '
\"	22	символ "
\?	3F	символ ?
\0	00	нульовий символ
\0ddd	-	вісімковий код символу
\0xddd	ddd	шістнадцятковий код символу

- рядкові - послідовності символів, заключених в подвійні лапки. Компілятор долучає в кінець рядка нуль-символ ‘\0’

Записи ‘c’ і “c” є різними.

Приклад: *"Це рядковий літерал!\n"*

“Національний університет\”КПІ\””

Коментарі

Види:

- однорядкові (Приклад: int a = 3; // *ініціалізація змінної a*)
- багаторядкові. Текст на Сі, що міститься у дужках /* та */ ігноруватиметься компілятором, тобто вважатиметься коментарем до програми. Приклад:

/* *функція обчислює суму матриць* */

Ключові слова

Ключові слова - це зарезервовані ідентифікатори, які мають спеціальне значення для компілятора.

auto	continue	float	interrupt	short	unsigned
asm	default	for	long	signed	void
break	do	far	near	sizeof	volatile
case	double	goto	pascal	static	while
cdecl	else	huge	switch	struct	
char	enum	if	register	typedef	
const	extern	int	return	union	

Структура програми

Основними частинами типової структури програми на Сі є такі:

- директиви препроцесорної обробки;
- опис зовнішніх змінних (вихідних даних і результатів) та функцій;
- функції програми;
- головна функція — програми **main()**.

```
1  /* Fig. 2.1: fig02_01.c
2     A first program in C */
3  #include <stdio.h>
4
5  int main()
6  {
7     printf( "Welcome to C!\n" );
8
9     return 0;
10 }
```

Функції

Функція – це синтаксично та логічно завершений самостійний фрагмент, що має ім'я та реалізує певну задачу.

Синтаксис

<тип_функції> <ім'я>([список параметрів])

{

< тіло функції>

}

Тіло – з описів операторів, кожен завершується **;**

Тип *void* - не повертає функція значення ОС,

int - повертає ціле число (0. якщо без помилок завершено програму)

Директиви препроцесора

#include <file_n>

Використання таких директив призводить до того, що препроцесор підставляє на місце цих директив тексти файлів у відповідності з тими, що перелічені у дужках < ... > .

Приклад:

```
#include <stdio.h>
        main()
{
                printf("Hello ! ...\n");
                return 0;
}
```

#define - макропідстановка

```
#define  N  20
```

Базові типи даних

Тип задає обсяг пам’яті для зберігання даних, визначає діапазон допустимих значень та встановлює операції, які можуть виконуватись. Типи: скалярні, агреговані (складені).

Базові типи даних Сі

1. **char** - символ
2. **int** - ціле
3. **float** - число з плаваючою комою одинарної точності
4. **double** - число з плаваючою комою подвійної точності

Дані дійсного типу представлені наближено. Точність представлення (десяткових цифр мантиси)

Float - 7 *double - 16*

Тип вказується явно в описах, тип констант – встановлюється за формою запису.

Приклад:

int I, k, letter ; double sum, result ; int m=10 , symb='' ;*

const double pi=3.14159

Тип	Діапазон значень	Розмір (байт)
char	-128 … 127	1
short		2
int	-32768 ... 32767	2 або 4
long	-2,147,483,648 ... 2,147,483,647	4
unsigned char	0 ... 255	1
unsigned short	0 … 65535	2
unsigned		2 або 4
unsigned long	0 ... 4,294,967,295	4
float	±(3.4 10-38 ... 3.4 1038)	4
double	±(1.7 10-308 ... 1.7 10308)	8
long double	±(3.4 10-4932 ... 3.4 104932)	10

Функції введення та виведення

Функція printf() призначена для виведення інформації за заданим форматом у стандартний вихідний потік (на екран).

Синтаксис функції printf():

printf("Рядок формату"[, аргумент1[, аргумент2, [...]]]);

Приклад :

```
#include<stdio.h>
  void main()
  {
          int a=10,b=20,c=30;
          printf(" a==%d \n b==%d \n c==%d \n",a,b,c);
  }
```

Специфікації повині бути узгоджені в порядку зліва направо із списком виведення.

Специфікації перетворення для функції printf():

- **%d** - десяткове ціле;
- **%f** - представлення величин float та double з фіксованою точкою;
- **%e** або **%E** - експоненціальний формат представлення дійсних величин;
- **%g** - представлення дійсних величин як f або E в залежності від значень;
- **%c** - один символ (char);
- **%s** - рядок символів;

Так, як і для функції **printf()**, для функції **scanf()** вказується рядок формату і список аргументів. Функція **printf()** використовує імена змінних, констант та вирази, в той час, як для функції **scanf ()** вказується тільки **покажчики** на змінні.

scanf("Рядок формату",&аргумент1[,&аргрумент2[, ...]]);

Кількість параметрів у списку введення повино відповідати кількості специфікацій, а тип змінних – бути сумісним з відповідними спеціфікаціями.

Функція **scanf ()** реалізує буферний принцип введення.

Приклад:

```
#include <stdio.h>
        void main()
        {
                int a,b,c;
                printf("A=");
                scanf("%d",&a);
                printf("B=");
                scanf("%d",&b);
                c=a+b;
                printf("A+B=%d",c);
        }
```