Лабораторна робота №7
Базові типи даних та перетворення типів даних

Мета роботи: отримання практичних навичок роботи з типами даних та перетворення типів в мові *C*.

Зміст:

Короткі теоретичні відомості

Базові типи даних мови C

Концепція типу даних з'явилася в мовах програмування високого рівня як природне віддзеркалення того факту, що дані, що обробляються програмою, можуть мати різні діапазони допустимих значень, зберігатися в пам'яті комп'ютера різним чином, займати різні обсяги пам'яті та оброблятися за допомогою різних команд процесора.

В *C* існує всього лише кілька базових типів:

- **char** – одиничний байт, який може містити один символ з допустимого символьного набору;

- **int** – ціле, зазвичай відображає цілочисельний регістр процесора;
- **float** – число з плаваючою точкою одинарної точності;
- **double** – число з плаваючою точкою подвійної точності.

Є також кілька кваліфікаторів, які можна використовувати разом із зазначеними базовими типами. Наприклад, кваліфікатори **signed** (зі знаком) або **unsigned** (без знаку) можна застосовувати до будь-якого цілочисельного типу. За замовчуванням змінні цілих типів мають знак, тому кваліфікатор **signed** є необов'язковим. Значення **unsigned** завжди невід'ємне.

Якщо кваліфікатори **short** (короткий) і **long** (довгий) застосовуються до цілих, то можна не вказувати тип **int**:

```
short a;            /* замість short int a;          */
long  b;            /* замість long  int b;          */
unsigned short c;   /* замість unsigned short int c; */
unsigned long  d;   /* замість unsigned long  int d; */
```

Нижче наведена таблиця з діапазонами значень базових типів з можливими кваліфікаторами:

Назва типу		Опис типу	Діапазон значень
Ціле	char	Ціле число	-128 … 127
	unsigned char	Ціле число без знаку	0 … 255
	short int	Ціле число	-32768 … 32767
	unsigned short int	Ціле число без знаку	0 … 65535
	int	Ціле число	-2^{31} … $2^{31}-1$
	unsigned int	Ціле число без знаку	0 … $2^{32}-1$
	long int	Ціле число	-2^{31} … $2^{31}-1$
	unsigned long int	Ціле число без знаку	0 … $2^{32}-1$
Дробові	float	Число з плаваючою точкою	$3.4\ 10^{-38}$ … $3.4\ 10^{+38}$
	double	Число з плаваючою точкою подвійної точності	$1.7\ 10^{-308}$ … $1.7\ 10^{+308}$

Оголошення змінних в мові *C*

Змінна – це іменована область пам'яті, до якої ми маємо доступ з програми; туди можна поміщати значення і потім отримувати їх. Кожна змінна мови *C* має певний тип, який характеризує розмір та розташування цієї області пам'яті, діапазон значень, які вона може зберігати, формат зберігання та набір операцій.

Мова *C* вимагає, щоб змінна була відома до першого звернення до неї. Це викликано необхідністю гарантувати правильність використання змінної відповідно до її типу.

Ім'я змінної, або ідентифікатор, може складатися з латинських букв, цифр і символу підкреслення. Великі і малі літери в іменах розрізняються. Крім того, ідентифікатор повинен починатися з букви або символу підкреслення, але не може починатися з цифри.

Числові константи в мові *C*

Коли в програмі зустрічається деяке число, наприклад 1, то це число називається константою. Константою тому, що ми не можемо змінити його значення. Кожна константа має певний тип, який визначається її значенням. Цілочисельні константи за умовчанням мають тип **int**, а константи з плаваючою точкою - тип **double**. Так, 0 має тип **int**, 3.14159 - тип **double**.

Константи цілих типів можна записати в десятковій, вісімковій та шістнадцятковій формі. Якщо константа починається з 0 (нуля), вона трактується як вісімкове число, якщо з 0x або 0X, то як шістнадцяткове. Звичний запис розглядається як десяткове число. Ось як виглядає число 25, записане як десяткова, вісімкова та шістнадцяткова константи:

```
25     // десяткова форма
031    // вісімкова форма
0x19   // шістнадцяткова форма
```

За замовчуванням всі цілі константи мають тип **int**.

Можна явно призначити цілій константі тип **long**, приписавши в кінці числа букву ***L*** (допускається як велика ***L***, так і мала ***l***, однак для зручності читання не слід вживати малу: її легко переплутати з одиницею).

Буква ***U*** (або ***u***) наприкінці визначає константу як **unsigned int**, а дві букви – ***UL*** або ***LU*** – як тип **unsigned long**.

Константи, що є дійсними числами, можуть бути записані як з десятковою точкою, так і в науковій (експоненціальній) формі запису. За умовчанням вони мають тип **double**. Для явної вказівки типу **float** потрібно використовувати суфікс ***F*** або ***f***, а для **long double** – ***L*** або ***l***, але тільки у разі запису з десятковою точкою.

Оператори

Оператори в програмі на мові *C* керують процесом виконання програми. У мові *C*, як і в інших мовах програмування, є ряд операторів, за допомогою яких можна організовувати повтори, вказувати інші оператори для виконання і передавати управління на іншу ділянку програми.

Оператор – це фраза алгоритмічної мови, що визначає закінчений етап обробки даних. До складу операторів входять ключові слова, дані, вирази та інші.

При виконанні програми на мові *C* її оператори виконуються в тому порядку, в якому вони з'являються в програмі, якщо немає оператора, який би явно передавав управління в інше місце програми.

Ті величини, над якими виконуються дії називаються операндами.

Прості і складені оператори

Найпростішою формою оператора є *порожній оператор*:

```
; /* порожній оператор */
```

Порожній оператор використовується там, де синтаксис мови вимагає вживання оператора, а логіка програми – ні. Випадкова поява зайвого порожнього оператора не викликає помилки компіляції. Наприклад, рядок

```
a = d + s;; /* немає помилки: зайвий порожній оператор */
```

складається з двох операторів – оператора додавання двох величин зі збереженням результату у змінній *a* і порожнього оператора.

Простий оператор складається з виразу, за яким слідує крапка з комою.

Наприклад:

```
int i = 1024; /* простий оператор визначення змінної */
```

Деякі конструкції мови синтаксично вимагають вживання одного оператора, пов'язаного з ними, тоді як логіка програми може вимагати виконання декількох. У таких випадках застосовуються *складені оператори* - послідовність простих, вміщена у фігурні дужки:

```
if ( ival0 > ival1 )
{
    /* складений оператор, що складається з
       оголошення і присвоювання */
    int temp = i;
    i = i1;
}
```

Складений оператор може вживатися там же, де і простий, і не потребує завершальної крапки з комою. Порожній складений оператор еквівалентний порожньому простому оператору.

Приведення (перетворення) типів у мові *C*

Якщо операнди оператора належать до різних типів, то при виконанні арифметичних операцій вони приводяться до деякого спільного типу. Приведення виконується відповідно з невеликим числом правил. Зазвичай, автоматично виконуються лише ті перетворення, які без будь-якої втрати інформації перетворюють операнди з меншим діапазоном значень в операнди з більшим діапазоном, як, наприклад, перетворення цілого в дробове у виразах на зразок **float + int**.

Вирази, які не мають сенсу, наприклад число з плаваючою точкою в ролі індексу, не допускаються.

Вирази, в яких могла б втрачатися інформація (скажімо, при присвоєнні довгих цілих змінним коротшим типам або при присвоєнні значень з плаваючою точкою цілим змінним), можуть спричинити попередження (***warning***), але вони допустимі.

Існує тонкість, що стосується перетворення символів в цілі числа: стандартом мови не визначено, чи є змінні типу **char** знаковими або беззнаковими. На деяких машинах значення типу **char** з одиничним старшим бітом буде перетворено на від'ємне ціле (за допомогою "поширення знака"). На інших - перетворення **char** в **int** здійснюється додаванням нулів зліва таким чином, що завжди отримується додатне значення.

Неявні арифметичні перетворення, як правило, здійснюються природним чином. У загальному випадку, коли такі оператори як + або * з двома операндами (бінарний оператор) мають різнотипні операнди, перш ніж операція почне виконуватися, "нижчий" тип (тип із меншим діапазоном) підвищується до "вищого" (тип із більшим діапазоном). Результат буде мати вищий тип.

Довідка. *Документація на правила перетворення типів у виразах знаходиться за словом promotion, зокрема для цілочисельних типів – integer promotions.*

Цілі і числа з плаваючою точкою

При перетворенні з типу з плаваючою точкою в цілочисельний, дробова частина значення відкидається; якщо отримане при цьому значення не можна представити в

заданому цілочисельному типі, то результат не визначений. Зокрема, не визначено результат перетворення негативних значень з плаваючою точкою в беззнакові цілі.

Якщо значення перетворюється з цілого у величину з плаваючою точкою і воно знаходиться в допустимому діапазоні, але подається в новому типі неточно, то результатом буде одне з двох значень нового типу, найближчих до вхідного. Наприклад, цілі числа типу **`long`** мають діапазон значень менший, ніж дійсні числа типу **`float`**. Проте число типу **`long`** може мати до 32 значащих розрядів, тоді як мантиса типу **`float`** має лише 23 біти. Таким чином, при присвоюванні цілочисельного значення дійсній змінній типу **`float`** може відбутися округлення. Якщо результат виходить за межі діапазону допустимих значень, поведінка програми не визначена.

Типи з плаваючою точкою

При перетворенні з типу з плаваючою точкою меншої точності в тип з плаваючою точкою більшої точності значення не змінюється. Якщо навпаки, перехід здійснюється від більшої точності до меншої і значення залишається в допустимих межах нового типу, то результатом буде одне з двох найближчих значень нового типу (округлене). Якщо результат виходить за межі діапазону допустимих значень, поведінка програми не визначена.

Явне перетворення типів

Явне перетворення типів може бути здійснене за допомогою операції приведення типів, яка має формат:

```
( ім'я-типу )операнд
```

Наприклад:

```
int a = 5;
```

```
float b;


b = (float) a; /* перетворення a цілого типу у дійсне значення */
```

Завдання для виконання

Створити новий проект, який містить програму для обчислення виразів *c1*, *c2*, *c3* та *c4* згідно варіанту (можна просто в медах одної програми в окремі фігурні дужки скопіювати завдання із 2-5 стовпчиків таблиці згідно варіанту). Результат обчислення величин вивести на екран (для цього використовувати функцію *printf()* із правильно підібраними ключами).

Пояснити отримані результати.

Номер варіанта	Завдання 1	Завдання 2	Завдання 3	Завдання 4

1	int a1 = 5;
	int b1 = 2;
	int c1;
	c1 = a1/b1;
		float a2 = 5;
		int c2;
		c2 = a2/2.0;
			unsigned char a3 = 100;
			unsigned short c3;
			c3 = a3 + 65540;
				unsigned char a4 = 100;
				unsigned short c4;
				c4 = a4 - 170;

2	int a = 5;
	int b = 2;
	float c1;
	c1 = a/b;
		float a2 = 5;
		int b2 = 2;
		float c2;
		c2 = a2/b2;
			unsigned char a3 = 100;
			unsigned char c3;
			c3 = a3 + 200;
				unsigned char a4 = 17;
				unsigned char c4;
				c4 = a4 - 20;

3	int a = 5;
	float c1;
	c1 = a/2.0;
		int a2 = 5;
		float b2 = 2;
		int c2;
		c2 = a2/b2;
			unsigned short a3= 100;
			unsigned char c3;
			c3 = a3 + 190;
				unsigned char a4 = 2;
				unsigned short c4;
				c4 = a4 -3;

4	int a = 4;
	int b = 2;
	int c1;
	c1 = (a+1)/b;
		int a2 = 5;
		double b2 = 2;
		int c2;
		c2 = a2/b2;
			unsigned char a3 = 100;
			unsigned char c3;
			c3 = a3 + 217;
				unsigned char a4 = 4;
				unsigned short c4;
				c4 = a4 - 170;

5	int a = 4;
	int b = 2;
	int c1;
	c1 = (a-1)/b;
		int a2 = 5;
		int c2;
		c2 = a2/2.0;
			unsigned short a3= 270;
			unsigned char c3;
			c3 = a3*2;
				unsigned char a4 = 4;
				unsigned char c4;
				c4 = a4 – 5;

6	float a = 4;
	float b = 2;
	int c1;
	c1 = (a+1)/b;
		double a2 = 5;
		int b2 = 2;
		int c2;
		c2 = a2/b2;
			unsigned char a3 = 199;
			unsigned char c3;
			c3 = a3*2;
				unsigned short a4=1000;
				unsigned char c4;
				c4 = a4 - 10;

7	float a = 4;
	float b = 2;
	int c1;
	c1 = (a-1)/b;
		int a2 = 5;
		int b2 = 2;
		float c2;
		c2 = a2/b2;
			unsigned char a3 = 200;
			unsigned short c3;
			c3 = a3*400;
				unsigned short a4=2000;
				unsigned char c4;
				c4 = a4 - 20;

8	float a = 5;
	int c1;
	c1 = a/2.0;
		Int a2 = 7;
		float c2;
		c2 = a2/2.0;
			unsigned char a3 = 17;
			unsigned char c3;
			c3 = a3*a3;
				unsigned short a4=10;
				unsigned char c4;
				c4 = a4 - 20;

9	int a = 5;
	int b = 2;
	int c = 2;
	int c1;
	c1 = a/b + c;
		float a2 = 7;
		int b2 = 2;
		float c2;
		c2 = a2/(b2+1);
			unsigned char a3 = 16;
			unsigned char c3;
			c3 = a3*(a3+5);
				unsigned char a4=18;
				unsigned char c4;
				c4 = a4 – 37;

10	int a = 5;
	int b = 2;
	float c = 2;
	int c1;
	c1 = a/b + c;
		int a2 = 7;
		int b2 = 2;
		float c2;
		c2 = a2/(b2+1);
			unsigned char a3 = 25;
			unsigned char c3;
			c3 = a3*(a3-3);
				unsigned short a4=1000;
				unsigned char c4;
				c4 = a4 - 100;

11	float a = 5;
	int b = 2;
	int c = 2;
	int c1;
	c1 = a/b + c;
		int a2 = 7;
		int b2 = 2;
		float c2;
		c2 = a2/(b2+1.0);
			unsigned short a3= 100;
			unsigned short c3;
			c3=a3*(a3+5)*(a3+10);
				unsigned short a4=2000;
				unsigned char c4;
				c4 = a4 - 200;

12	int a = 5;
	float b = 2;
	int c = 2;
	int c1;
	c1 = a/b + c;
		int a2 = 8;
		int b2 = 2;
		float c2;
		c2 = (a2+1)/b2+1;
			unsigned short a3= 150;
			unsigned char c3;
			c3 = a3 + 217;
				unsigned char a4=100;
				unsigned char c4;
				c4 = a4 - 140;

13	int a = 5; int b = 3;	float a2 = 8; int b2 = 2;	unsigned char a3 = 80; unsigned char c3;	unsigned short a4=1; unsigned char c4;

	int c = 1;
	int c1;
	c1 = a/b - c;
		float c2;
		c2 = (a2+1)/b2+1;
			c3 = a3*(a3-4);	c4 = a4 - 2;

14	int a = 5;
	int b = 3;
	int c = 1;
	float c1;
	c1 = a/b - c;
		int a2 = 8;
		int b2 = 2;
		int c2;
		c2=(a2+1)/b2+1.5;
			unsigned char a3 = 180;
			unsigned char c3;
			c3 = a3*3;
				unsigned char a4=1;
				unsigned short c4;
				c4 = a4 - 2;

15	float a = 5;
	int b = 3;
	int c = 1;
	float c1;
	c1 = a/b - c;
		int a2 = 8;
		int b2 = 2;
		float c2;
		c2 = (a2+1)/b2+1;
			unsigned char a3 = 180;
			unsigned char c3;
			c3 = (a3+7)*2;
				unsigned char a4=100;
				unsigned short c4;
				c4 = a4 - 200;

16	int a = 7;
	int b = 2;
	int c = 1;
	int c1;
	c1 = a/(b + c);
		int a2 = 8;
		int b2 = 2;
		float c2;
		c2 = (a2+1.0)/b2;
			unsigned char a3 = 250;
			unsigned short c3;
			c3 = a3*(a3+25)+3;
				unsigned char a4=100;
				unsigned char c4;
				c4 = a4 - 200;

17	int a = 7;
	int b = 2;
	int c = 1;
	float c1;
	c1 = a/(b + c);
		int a2 = 10;
		int c2;
		c2 = a2/(a2 - 6);
			unsigned char a3 = 250;
			unsigned char c3;
			c3 = a3+25;
				unsigned short a4=1;
				unsigned short c4;
				c4 = a4 - 2;

18	float a = 7;
	int b = 2;
	int c = 1;
	int c1;
	c1 = a/(b + c);
		int a2 = 10;
		float c2;
		c2=(a2+4.0)/(a2-6);
			unsigned char a3 = 200;
			unsigned char c3;
			c3 = a3+100;
				unsigned short a4=1000;
				unsigned char c4;
				c4 = a4 - 1;

19	int a = 7;
	int c1;
	c1 = a/3;
		int a2 = 10;
		int c2;
		c2=(a2+4.0)/(a2-6);
			unsigned char a3 = 200;
			unsigned char c3;
			c3 = a3*3;
				unsigned short a4=700;
				unsigned char c4;
				c4 = a4 - 1;

20	int a = 7; int c1;	int a2 = 7; int b2 = 2;	unsigned short a3= 200; unsigned char c3;	unsigned short a4=12; unsigned char c4;

	c1 = a/3.5;	int d2 = 1; int c2; c2 = a2/b2 – d2;	c3 = a3*3;	c4 = a4 – 700;

21	int a = 7;
	float c1;
	c1 = a/3;
		int a2 = 5;
		int b2 = 3;
		int d2 = 1;
		float c2;
		c2 = a2/b2 – d2;
			unsigned short a3= 500;
			unsigned short c3;
			c3 = a3*(a3+7);
				unsigned short a4=10;
				unsigned char c4;
				c4 = a4 – 17;

Контрольні питання

1. Які базові типи даних в мові С ви знаєте?
2. Скільки байт відводиться для зберігання кожного з базових типів даних в мові С?
3. З яких компонент складається оголошення змінної в мові С?
4. У чому відмінність простих і складених операторів?
5. Як відбувається перетворення з типів плаваючою точкою в цілочисельні?
6. Як відбувається перетворення з цілочисельні типів в типи з плаваючою точкою?