Тема №16

Типове данни. Оператор за присвояване. Изрази

1. Величини.

Всички данни, които програмата обработва се наричат с общото име величини.

а) константи;

Те имат точно определена стойност, която не се променя по време на изпълнение на програмата, а само се използва в процеса на изчисление.

б) променливи.

Техните стойности се променят по време на изпълнение на програмата. Стойността на променливата в определен момент се нарича нейна текуща стойност.

Имената на константите и променливите е добре да се избират така, че да напомнят за смисъла им. Те могат да бъдат записани с латински букви, цифри и долно тире, но не могат да започват с цифри. Освен това C++ прави разлика между главни и малки букви. Например ABC, Abc, aBC, abc са имена на различни променливи.

2. Тип на данните.

Тип на данните – характеристика, определяща какви могат да бъдат конкретните стойности на величините и кои са характерните операции, които могат да се извършват с тях.

Всеки тип може да се разглежда като съвкупност на три множества:

- Множество от стойности, които могат да приемат величините от типа;
- Множество от операциите, в които могат да участват величините от типа;
- Множество от отношения (релации) между величините от типа.

Преди да бъде използвана всяка променлива трябва да бъде декларирана чрез посочване на типа й пред нейното име.

3. Основни типове данни.

Най-общо можем да разделим типовете данни в C++ на две групи:

- Скаларни типове данни – булев, целочислен, реален, символен, изброен, указател, псевдоним;
- Съставни типове данни – масив, символен низ, вектор.

Ние ще използваме само част от тях.

а) Целочислени типове данни;

- ***short*** – цели числа от -32 768 до 32 767 (дължина – 2 байта);
- ***int*** – цели числа от -2 147 483 648 до 2 147 483 647 (дължина – 4 байта);

б) Реални типове данни;

Реалните числа могат да се записват по три начина: с десетична точка; без десетична точка, но с експонентна част; с десетична точка и с експонентна част.

- ***float*** – числа с плаваща точка от -3.4E38 до -3.4E-38 и от от 3.4E-38 до 3.4E38 (дължина – 4 байта);
- ***double*** – числа с плаваща точка с двойна точност от -1.7E308 до -1.7E-308 и от от 1.7E-308 до 1.7E308 (дължина – 8 байта);

в) Логически (булев) тип данни;

- ***bool*** – лъжа (false) – 0; истина (true) – 1 (различно от 0)

г) Символен тип данни;

- ***char*** – знак

Символите се ограждат с апострофи: 'a', '2', '?'.

Символът апостроф се изобразява чрез наклонена черта: '\''.

Някои символи имат специално предназначение: '\a' – издава звуков сигнал; '\b' – връща курсора с един символ назад; '\r' – връща курсора в началото на реда; '\n' – предизвиква преминаване на нов ред.

Преобразуването от един тип в друг, ако е възможно, може да се направи със задаване на заградения в скоби нов тип пред израза.

Напр. (int) t – преобразува t в цяло число от тип integer

4. Оператор за присвояване.

Операторът за присвояване има следния вид:

<променлива>=<израз>;

Променливата е идентификатор, дефиниран вече като променлива. Изразът е от тип, съвместим с типа на променливата.

Намира се стойността на израза. Ако тя е от тип, различен от типа на променливата, конвертира се, ако е възможно, до него и се записва в паметта, отделена за променливата. Това е неявен начин да се преобразува типът.

Ако променливата е от тип bool, изразът може да бъде от тип bool или от който и да е числов тип. Ако променливата е от тип double, всички числови типове, а също типът bool, може да са типове на израза.

В езика за програмиране C++ са въведени някои съкратени форми на оператора за присвояване. Например заради честото използване на оператора: a=a+1; той съкратено се означава така: a++;. Въведено е и означението a--; на оператора a=a-1;.

Използват се и съставни оператори за присвояване. Напр. a+=2 е еквивалентно на a=a+2; a-=2 (a=a-2); a*=2 (a=a*2); a/=2 (a=a/2); a%=2 (a=a%2).

5. Оператор за вход.

Операторът за вход има следния вид:

cin >> <променлива>;

където cin означава, че се въвеждат от клавиатурата данни и се присвояват на променливата, а самата променлива е идентификатор, дефиниран от "допустим тип" преди оператора за въвеждане.

Операторът >> се изпълнява отляво надясно. Затова, изразът:

cin >> променлива1 >> променлива2 >> … >> променливаN ;

е еквивалентен на редицата от изрази:

cin >> променлива1 ;
cin >> променлива2 ;
……
cin >> променливаN ;

Изпълнението на:

cin >> променлива1 >> променлива2 >> … >> променливаN ;

предизвиква пауза. Очаква се да бъдат въведени N стойности – за посочените променливи съответно и да бъде натиснат клавишът ENTER. Тези стойности трябва да бъдат въведени по подходящ начин (на един ред, на отделни редове или по няколко данни на последователни редове, слепени или разделени с интервали, табулации или знака за нов ред), като стойностите трябва да бъдат от тип, съвместим с типа на съответната променлива.

6. Оператор за изход.

Операторът за изход има следния вид:

cout << <израз>;

Операторът << извежда върху екрана на компютъра, стойността на израза, който трябва да е от допустим тип (такива типове са: bool, int, double и др.). Чрез верига от оператори могат да бъдат изведени стойностите на повече от един израз, като се изпълняват отляво надясно.

Изразът: cout << израз1 << израз2 << … << изразN ;

е еквивалентен на редицата от изрази:

cout << израз1 ;

cout << израз2 ;

……

cout << изразN ;

Пример: cout << "a= " << a << "\n " ;

Тук записът "\n " означава, че след извеждане на стойността на променливата a ще се премине на нов ред. Преминаването на нов ред може да стане и чрез endl (cout << "a= " << a << endl;)

За да могат да се използват операторите за вход и изход е необходимо да се използва библиотеката iostream, което става, като се запише в началото на програмата: #include <iostream>.

7. Манипулатори за изход.

- **setw (<цял израз>)** – посочва броя позиции, в които да се помести следващана за извеждане стойност

Следващият пример ще изведе: -----12345-12345 (интервалите са заместени с тирета):

int r=12345

cout << setw(10) << r << endl;

- **setprecision (<цял израз>)** – посочва броя цифри, които да се използват при извеждане на следващите реални числа, а в съчетание с setiosflags(ios::fixed) – задава броя на цифрите след десетичната точка на извежданите реални числа.

Следващият пример ще изведе на единия ред: -----23.47, а на другия: ---23.4688.

double r=23.468792;

cout << setprecision(4) << setw(10) << r << endl;

cout << setiosflags(ios::fixed) << setprecision(4) << setw(10) << r << endl;

За да се използват манипулатори, трябва да се добави в програмата библиотеката iomanip.

8. Математически и логически операции.

Знак	***Значение***	***Знак***	***Значение***
+	събиране	**>**	по-голямо
-	изваждане	**<**	по-малко
*****	умножение	**>=**	по-голямо или равно
/	деление	**<=**	по-малко или равно
%	остатък при целочислено деление	**!**	логическо "не" (NOT)
==	равно	**&&**	логическо "и" (AND)
!=	различно	**||**	логическо "или" (OR)

9. Приоритет на операциите.

1	**()**
2	**!**
3	**(тип)**
4	*** / %**
5	**+ -**
6	**< <= > >=**
7	**== !=**
8	**&& ||**
9	**= += -= *= /= %=**

10. Някои стандартни функции

1	**abs(x)**	Връща абсолютната стойност на x, x – целочислено
2	**fabs(x)**	Връща абсолютната стойност на x, x – реално
3	**sqrt(x)**	Връща квадратен корен от x, x – неотрицателно
4	**pow(x,n)**	Връща x^n, x и n – реални от тип double
5	**sin(x)**	Връща синус от ъгъл x, зададен в радиани
6	**cos(x)**	Връща косинус от ъгъл x, зададен в радиани

7	**tan(x)**	Връща тангенс от ъгъл x, зададен в радиани
8	**exp(x)**	Връща e на степен x (e^x)
9	**log(x)**	Връща натурален логаритъм (ln x) от x, x - положително
10	**log10(x)**	Връща десетичен логаритъм от x, x - положително
11	**ceil(x)**	Връща най-малкото цяло число, по-голямо или равно на x, преобразувано в тип double
12	**floor(x)**	Връща най-голямото цяло число, по-малко или равно на x, преобразувано в тип double

Вградените математически функции се намират в библиотеката math.h (т. е. необходимо е да се запише #include <math.h>).

11. Пример за програма на C++.

```
#include <iostream>
using namespace std;
int main()
 {
    const double PI=3.14159265;
    double C,S,R;
    cout << "Vyvedete radius: ";
    cin >> R;
    C=2*PI*R;
    S=PI*R*R;
    cout << "Dylzhinata na okryzhnost s radius " << R << " e " << C << endl;
    cout << "Liceto ns kryg s radius " << R << " e " << S << endl;
    system("pause");
    return 0;
 }
```

Най-напред програмата започва с включването на използваната библиотека iostream. Вторият ред указва, че ще се използват имена, определени в т. нар. област std (това са cin, cout, endl). Главната програма представлява функция с фиксираното име main. В нея най-напред е декларирана константата PI, както и променливите C, S и R.

В езика C++ cin е име на специална променлива, определена предварително, и означава входен поток. Можем да си представим входния поток като редица от знаци, който тръгва от стандартното входно устройство и е насочен към променливата, посочена като втори аргумент на операцията за вход >>. По подобен начин cout е предварително дефинирана и означава изходен поток от знаци, който е насочен към стандартното изходно устройство. В случая това е мониторът. Операцията << е двуаргументна. Нейният първи аргумент е cout, а вторият в общия случай е някакъв израз. Всъщност cin и cout не са точно променливи, а обекти съответно от класовете istream и ostream, но това засега ще остане извън нашето внимание.

В нашата програма най-напред се извежда подканващо съобщение да напишем радиус. Всеки текст, ограден в кавички се приема за текстова константа и ще се изведе точно във вида, в който е написана. После се въвежда стойността за радиус с оператора за въвеждане. Следва присвояване на получените чрез математическите изрази стойности на дължината на окръжността и лицето на кръга. И накрая се извеждат получените стойности. Манипулаторът endl указва края на текущия ред за отпечатване и оптечатването на следващия текст (ако има такъв) ще се извърши на нов ред.

System("pause") ще позволи да не се затвори веднага след показването прозорецът с резултата, докато не натиснем клавиш. Накрая главната функция трябва да върне резултат, затова се задава return 0.