Лекція 11-12

Операції вводу-виводу. С++ введення виведення потоками. Функції вводу-виводу С С++

Почнемо розглядати процедури вводу виводу даних із стандартних функцій базової мови С.

Бібліотека

stdio.h. – С**,** **cstdio** – С++

stdio.h (від англ. standard input/output header) – заголовок стандартної бібліотеки мови Сі, що складається з визначень макросів, констант і оголошення функцій і типів, які використовуються для різних операцій стандартного введення і виведення.

Розглянемо деякі функції цієї бібліотеки:

Функції, макроси	Призначення
printf()	виведення інформації у стандартний потік (через буфер на укран)
scanf()	зчитування інформації із стандартного потоку (через буфер з екрану)
int getc(), int getchar	зчитує з екрану або файлу і повертає символ з даного потоку і змінює покажчик позиції файлу
putc(char),putchar()	записує і повертає символ в потік і змінює покажчик позиції файлу на нього
char *gets(char *str)	зчитує символьний рядок
puts()	виводить символьний рядок

Функція ***printf***

- printf ("управляючий рядок", список змінних) – для виведення інформації

Для функції виводу управляючий рядок включає:

- керуючі символи;
- текст, представлений для безпосереднього виведення;
- формат, призначений для виведення значень змінних різних типів.

Якщо виводиться тільки текст, який зазначений в управляючому рядку, то список змінних може бути відсутній.

Розглянемо компоненти управляючого рядка, які обов’язково беруться у лапки.

Керуючі символи не виводяться на екран, а керують розташуванням символів, що виводяться. Відмінною рисою керуючого символу є наявність зворотного слеша '\' перед ним.

Основні керуючі символи:

'\ n' - новий рядок;

'\ t' - горизонтальна табуляція;

'\ v' - вертикальна табуляція;

'\ b' - повернення на символ;
'\ r' - повернення на початок рядка;
'\ a' - звуковий сигнал.

Формати потрібні для того, щоб вказувати вид, в якому інформація буде виведена на екран. Відмінною рисою формату є наявність символу відсоток '%' перед ним:
% d – ціле число типу int зі знаком в десятковій системі числення;
% u – ціле число типу unsigned int;
% x – ціле число типу int зі знаком в шістнадцятковій системі числення;
% o – ціле число типу int зі знаком в вісімковій системі числення;
% hd – ціле число типу short зі знаком в десятковій системі числення;
% hu – ціле число типу unsigned short;
% hx – ціле число типу short зі знаком в шістнадцятковій системі числення;
% ld – ціле число типу long int зі знаком в десятковій системі числення;
% lu – ціле число типу unsigned long int;
% lx – ціле число типу long int зі знаком в шістнадцятковій системі числення;
% f – дійсний формат (числа з плаваючою точкою типу float);
% lf – дійсний формат подвійної точності (числа з плаваючою точкою типу double);
% e – дійсний формат в експоненційної формі (числа з плаваючою точкою типу float в експоненційної формі);
% c – символьний формат;
% s – строковий формат.

В управляючому рядку символи формату визначають тип змінної, яка виводиться. Функція працює наступним чином: замість символу формату виводиться значення відповідної змінної із списку.

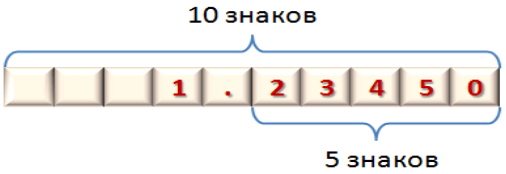
Приклад.

```
#include <stdio.h>
int main()
{ int a = 5;
  float x = 2.78;
  printf("a=%d\n", a);
  printf("x=%f\n", x);
  getchar(); getchar();
  return 0;}
```

Результат роботи програми:
a=5
x=2.780000

При вказанні формату можна явно вказати загальну кількість знакомісць і кількість знакомісць, займаних дробової частиною. Загальна кількість знакомісць для дійсного числа включає знак числа і десяткову крапку.

Наприклад, формат %10.5f представлений нижче.



Приклад.

Для вище наведеного прикладу запишемо функцію виводу:

```
printf("a=%2d x=%5.2f\n",a x);
```

Результат виконання: a=5 x=2.78

Функція ***printf*** є універсальним засобом виводу але її недоліком є те, що вона не дозволяє напряму управляти кольором та позиціонуванням. При її використанні ці операції можна виконувати лише за допомогою Esc-послідовностей.

Для введення інформації в бібліотеці **stdio.h** передбачена функція ***scanf***:

```
scanf ("Управляюча строка", список змінних...);
```

Управляюча строка функції містить формати, за якими вводяться значення змінних із списку. На відміну від функції ***printf*** список змінних для функції ***scanf*** містить їх адреси. Для формування адреси використовується операція &.

Приклад.

```
#include <stdio.h>
int main()
{ float y;
  printf("Введите y: "); // выводим сообщение
  scanf("%f", &y);     // вводим значения переменной y
  printf("Значение переменной y=%f", y); // выводим значение переменной y
  getchar(); getchar();
  return 0;}
```

Макрос ***getc()*** аналогічний до функції ***getchar()*** повертає один символ. Для виводу символу використовують відповідно ***putc()*** або ***putchar()***.

Для зчитування та виводу рядка символів використовують відповідно функції ***gets()*** та ***puts()***

Приклад. Наведена програма зчитує та виводить символи, набрані на клавіатурі, поки не буде введений символ крапки.

```
#include <cstdio>

int main ()
{ char character;
  puts("Введите символ, символ точки - выход('.'):");
  do  {
    character = getchar(); // считать введённый со стандартногопотока  ввода
символ
    putchar (character);   // вывести этот символ
  } while (character != '.'); // пока введенный символне точка
  return 0; }
```

Бібліотека conio.h

Функції ***printf*** та ***scanf*** є універсальними засобом виводу-вводу але її недоліком є те, що вони не дозволяють напряму управляти кольором та позиціонуванням, так як операції виконуються через буфер. При їх використанні, ці операції можна виконувати лише за допомогою Esc-послідовностей.

Для реалізації прямого вводу виводу використовують функції бібліотеки **conio.h**, а саме функції ***cprintf*** та ***cscanf***, ***getch()*** та ***putch()***. Синтаксис функцій аналогічний до функцій ***printf***, ***scanf***. Але бібліотека **conio.h** включає спеціальні функції управляння екраном. Розглянемо деякі основні функції.

Функція	**Призначення**
clrscr(void)	очистка екрану
clreol(void)	очистка поточного рядка
delline(void)	видалення поточного рядка
gotoxy(int column, int row)	встановлення курсору у задану позицію на екрані
int wherex(void) int wherey(void)	визначення поточної позиції курсора на екрані
void insline(void)	вставка пустого рядка
int gettext(int left, int top, int right, int bottom, void *buffer)	копіювання тексту з екрана у буфер
int puttext(int left, int top, int right, int bottom, void *buffer)	виведення тексту з буфера на екран
textcolor(color)	встановлення кольору тексту
textbackground(int color)	встановлення кольору екрану
void highvideo(void) void lowvideo(void) void normvideo(void)	управління яскравістю
void _setcursortype(int type)	вигляд курсору1

Таблиця кольорів

Колір	Константа	Значення	Призначення
Чорний	BLACK	0	текст екран
Синій	BLUE	1	
Зелений	GREEN	2	
Бірюза	CYAN	3	
Червоний	RED	4	
Фіолетовий	MAGENTA	5	
Коричневий	BROWN	6	
Яскраво-сірий	LIGHTGRAY	7	
Темно-сірий	DARKGRAY	8	текст
Яскраво-синій	LIGHTBLUE	9	

Яскраво-зелений	LIGHTGREEN	10	
Яскраво-бірюзовий	LIGHTCYAN	11	
Яскраво-червоний	LIGHTRED	12	
Яскраво-фіолетовий	LIGHTMAGENTA	13	
Жовтий	YELLOW	14	
Білий	WHITE	15	
Блимання	BLINK	128	

Приклади.

1. Визначення позиції курсору:

```
#include <conio.h>

        int xpos, ypos;
        xpos = wherex();
        ypos = wherey();  }
```

2. Переміщення тексту

```
#include <conio.h>

void main(void) {
{char buffer[128];
int row, column;

clrscr();
cprintf("Мова програмування C\n\r");
gettext(1,1,24,1,buffer);
clrscr();
row=2; column=20;
puttext(column,row,colbmn+22,row,buffer);   }
```

Функція ***int getch(void)*** повертає черговий символ, лічений з консолі, але не виводить цей символ на екран.

Функція ***int getche(void)*** повертає черговий символ, лічений з консолі, і виводить цей символ на екран.

Жодна з цих функцій не визначена стандартом ANSI C.

Функція ***int putch (int c)*** виводить символ *c* в консоль без буферизації. У разі успіху повертає *c*. В іншому випадку повертає *EOF*.

C++ введення виведення потоками

Бібліотека потокового введення-виведення **iostream** може бути використана в якості альтернативи відомої стандартної бібліотеки вводу-виводу мови С - **stdio**. Для нас бібліотека **iostream** цікава як прекрасний приклад об'єктно-орієнтованого проектування, так як містить багато характерних прийоми і конструкції.

В основі ООП підходу, реалізованого засобами **iostream**, лежить припущення про те, що об'єкти мають знання того, які дії слід робити при введенні і виведенні.

При користуванні бібліотекою **iostream** помилки, пов'язані з "змішування" типів даних, виключені. Якщо ви використовуєте в операції

введення-виведення змінну типу ***unsigned long***, то викликається підпрограма, відповідальна саме за цей тип.

Бібліотека **stdio** підтримує засоби мови С, що дозволяє використовувати змінне число параметрів. Але така гнучкість дається не даром – на етапі компіляції перевірка відповідності між специфікацією формату, як у функціях ***printf () і scanf (),*** не виконується.

У бібліотеці **iostream** застосований інший підхід. Оператори введення-виведення оформляються у вигляді виразів із застосуванням переобумовлених функцій-операторів для кожного з типів даних, що зустрічаються в виразі. Якщо вам необхідно використовувати новий тип даних в операціях введення-виведення, ви можете розширити бібліотеку iostream своїми функціями-операторами.

Бібліотека **iostream** повільніша, ніж **stdio**, але це невелика плата за надійність і розширюваність, що базуються на можливостях об'єктно-орієнтованих засобів виведення.

В бібліотеці **iostream** визначено два типи: **istream** і **ostream**. **istream** представляє потік введення, а **ostream** – потік виведення.

Термін "потік" в даному випадку представляє послідовність символів, яка записується на пристрій вводу-виводу або зчитується з нього. І в даному випадку під пристроєм введення-виведення розглядається консоль.

Для вводу інформації застосовується функція:

cin >> список змінних;

для виводу:

cout<< список змінних;

Також передбачений стандартний потік ***cerr*** для виводу повідомлень про помилки.

Наприклад, якщо змінна *x* описана як цілочисельна, то команда ***cin >> x***; означає, що в змінну *x* буде записано певне ціле число, введене з клавіатури. Якщо необхідно ввести декілька змінних, то слід написати ***cin >> x >> y >> z;***.

Об'єкт ***cout*** і операція << дозволяє вивести на екран значення будь-якої змінної або текст. Текст необхідно укладати в подвійні лапки. Запис ***cout*** << ***x***; означає виведення на екран значення змінної *x*.

Для того, що б закрити вихідний потік на кінці команди ***cout*** необхідно позначити кінець рядка ключовим словом ***endl***.

```
cout << "Программа стартовала"<< endl;
```

Для управління потоками в мові С++ передбачені маніпулятори.

Таблиця основних маніпуляторів

Маніпулятор	Опис
endl	Помещение в выходной поток символа конца строки '\n'
dec	Установка основания 10-ой системы счисления
oct	Установка основания 8-ой системы счисления

Маніпулятор	Опис
hex	Установка основания 16-ой системы счисления
setbase	Вывод базовой системы счисления
width(ширина)	Устанавливает ширину поля вывода
fill('символ')	Заполняет пустые знакоместа значением символа
precision(точность)	Устанавливает количество значащих цифр в числе (или после запятой) в зависимости от использования fixed
fixed	Показывает, что установленная точность относится к количеству знаков после запятой
showpos	Показывает знак + для положительных чисел
scientific	Выводит число в экспоненциальной форме
get()	Ожидает ввода символа
getline(указатель, количество)	Ожидает ввода строки символов. Максимальное количество символов ограничено полем количество

Приклад. Введення-виведення даних

C++

```
#include <iostream>
using namespace std;
int main()
{ int n;
  cout << "Введите n:";
  cin >> n;
  cout << "Значение n равно: " << n << endl;
  cin.get(); cin.get();
  return 0;
}
```

C

```
#include <stdio.h>
int main()
{
  int n;
  printf("Введите n:");
  scanf("%d", &n);
  printf("Значение n равно: %d\n", n);
  getchar(); getchar();
  return 0;
}
```

Приклад. Форматоване введення-виведення в C++

```
#include <iostream>
using namespace std;
int main()
{
  double a = -112.234;
  double b = 4.3981;
  int c = 18;
  cout << endl << "double number:" << endl;
  cout << "width(10)" << endl;
  cout.width(10);
  cout << a << endl << b << endl;
  cout << "fill('0')" << endl;
  cout.fill('0');
  cout.width(10);
```

```
  cout << a << endl << b << endl;
  cout.precision(5);
  cout << "precision(5)" << endl << a << endl << b << endl;
  cout << "fixed" << endl << fixed << a << endl << b << endl;
  cout << "showpos" << endl << showpos << a << endl << b << endl;
  cout << "scientific" << endl << scientific << a << endl << b << endl;
  cout << endl << "int number:" << endl;
  cout << showbase << hex << c << " " << showbase << oct << c << " ";
  cout << showbase << dec << c << endl;
  cin.get();
  return 0;
}
```

Результат роботи програми:

```
double number:
width(10)
  -112.234
4.3981
fill('0')
00-112.234
4.3981
precision(5)
-112.23
4.3981
fixed
-112.23400
4.39810
showpos
-112.23400
+4.39810
scientific
-1.12234e+02
+4.39810e+00

int number:
0x12 022 +18
```

В середовищі Visual Studio для запису або виведення символів на консоль застосовується об'єкт **cout**, який представляє тип **ostream**, а для читання з консолі використовується об'єкт **cin** класу **std**.

Належність класу позначається «::»

Приклад.

```
#include <iostream>
using namespace std;

int main()
{  int age;
   double weight;
   std::cout << "Input age: ";
   std::cin >> age;
   std::cout << "Input weight: ";
   std::cin >> weight;
   std::cout << "Your age: " << age << "\t your weight: " << weight << std::endl;
   return 0;}
```

Результат роботи програми:

```
Input age: 32
Input weight: 67.45
Your age: 32       your weight: 67.45
```