Вход и изход в езика С
Входно изходни потоци
Входно изходни функции

Базово програмиране
Лекция 3
Милена Карова
кат. КНТ ТУ-Варна

Входно/изходни потоци

- Целият вход/изход на C/C++ програмата е организиран чрез потоци.
- Поток (stream) набор от ASCII символи, организирани в редове като програмата не се интересува, кой е източника или приемника на потока
- Съществуват 3 потока: input stream, output stream и standard error, свързан към екрана (съобщенията за грешки се извеждат към този поток)

Извеждане на данни чрез cout<<

- cout

cout<<number_of_bars;

- cout е стандартния обект към който операция << извежда данните. В началния случай cout е свързан със стандартното входно у-во клавиатура.
- Възможно е да се извеждат константи, променливи, изрази и стойностите на други функции. Следните 2 примера са идентични

```
cout << number_of_bars << " candy bars\n";
cout << one_weight << " ounces each\n";
```

```
cout << number_of_bars << " candy bars\n" << one_weight  << " ounces each\n";
```

Форматиран изход

- Всички стрингови низове, които се извеждат трябва да бъдат в кавички
- \n означава „мини на нов ред“. Може да се замести от изходния манипулатор **endl**
- *cout<< numbers_of_bars<<“candy bars”<<endl<< one_weght<<“ounces each”<<endl;*
- \ показва, че последвалият символ не е точно символа, а управляваща последователност (escape sequence)
- При използването на \ в стринг трябва да се удвои \\
- Други: \t табулатор, \“двойни кавички и др.

Форматиран изход

- Функция printf()
  - Необходимост от файла stdio.h
  - Закръгля реалните числа до показан брой цифри след десетичната запетая
  - Извежда ляво и дясно подравнени данни
  - Представя числовите данни в експоненциален формат
  - Представя шестнадесетични числа
  - Извежда данните с необходимата точност и с необходимата ширина на полетата

```
printf("Using Source IP: %s port: %u\n", argv[1], atoi(argv[2]));
printf("\t%.3f\n\t%.3e\n\t%.3g\n\n", f, f, f);
```

- Примерът е за цели стойности

Conversion specifier	Description
d	Display a signed decimal integer
i	Display a signed decimal integer. (Note: The i and d specifiers are different when used with scanf().)
o	Display an unsigned octal integer.
u	Display an unsigned decimal integer.
x or X	Display an unsigned decimal integer. x causes the digit 0-9 and the letters A-F to be displayed, and x causes the digits 0-9 and a-f to be displayed.
h or l (letter l)	Place **before** any integer conversion specifier to indicate that a short or long integer is displayed respectively.

Форматиран изход

- За числа с плаваща запетая (реални)

Conversion specifier	Description
`e` or `E`	Display a floating-point value in exponential notation.
`f`	Display floating-point values.
`g` or `G`	Display a floating-point value in either the floating-point form `f` or the exponential form `e` (or `E`)
`l`	Place before any floating-point conversion specifier to indicate that a `long double` floating-point value is displayed.

- Символи и стрингове: %c за символи и %s за низове

```
//Printing strings and characters
#include <stdio.h>
#include <stdlib.h>

int main()
{
char character = 'A';
char string[]  =  "This is a string";
char *stringPtr = "This is also a string";
printf("---------------------------------\n");
printf("---Character and String format---\n");
printf("---------------------------------\n\n");
printf("%c <--This one is character\n", character);
printf("\nLateral string\n");
printf("%s\n", "This is a string");
printf("\nUsing array name, the pointer to the first array's element\n");
printf("%s\n", string);
printf("\nUsing pointer, pointing to the first character of string\n");
printf("%s\n", stringPtr);
system("pause");
return 0;
}
```

Escape последователности

Escape sequence	Description
\'	Output the single quote (') character.
\"	Output the double quote (") character.
\?	Output the question mark (?) character.
\\	Output the backslash (\) character.
\a	Cause an audible (bell) or visual alert.
\b	Move the cursor back one position on the current line.
\f	Move the cursor to the start of the next logical page.
\n	Move the cursor to the beginning of the next line.
\r	Move the cursor to the beginning of the current line.
\t	Move the cursor to the next horizontal tab position.
\v	Move the cursor to the next vertical tab position.

printf или cout<<

- C

```
printf("Enter new tag:  ");
scanf("%d", &tag);
printf("The new tag is: %d\n", tag);
```

- C++

```
cout<<"Enter new tag: ";
cin>>tag;
cout<<"The new tag is: "<<tag<<'\n';
```

- Не се изисква форматиращ стринг и спецификатори
- Операторът & от scanf не се използва при въвеждане
- Iostream библиотеката съдържа стотици възможности: cin, cout, cerr(небуфериран изход при грешки) и clog(буфериран стандартен изход за грешки)

Форматиране на числа с десетична точка

- Числата се изписват с десетична точка (няма десетична запетая
- Експоненциален или scientific format означава степен с основа на степента 10:
  - 0.000000000001=0,1x10^{-11} записва се 0.1E-11
  - 1000000000000=1x10^{12} записва се 1E+12
  - Този формат е характерен за много големи и много малки числа
- Ако double променливата price има стойност 78.5 то следните команди са валидни:

```
cout << "The price is $" << price << endl;
```

```
The price is $78.500000
```

```
cout.setf(ios::fixed);
cout.setf(ios::showpoint);
cout.precision(2);
cout << "The price is $" << price << endl;
```

```
The price is $78.50
```

Форматиращи флагове за setf

Flag	Meaning	Default
`ios::fixed`	If this flag is set, floating-point numbers are not written in e-notation. (Setting this flag automatically unsets the flag `ios::scientific`.)	Not set
`ios::scientific`	If this flag is set, floating-point numbers are written in e-notation. (Setting this flag automatically unsets the flag `ios::fixed`.) If neither `ios::fixed` nor `ios::scientific` is set, then the system decides how to output each number.	Not set
`ios::showpoint`	If this flag is set, a decimal point and trailing zeros are always shown for floating-point numbers. If it is not set, a number with all zeros after the decimal point might be output without the decimal point and following zeros.	Not set
`ios::showpos`	If this flag is set, a plus sign is output before positive integer values.	Not set
`ios::right`	If this flag is set and some field-width value is given with a call to the member function `width`, then the next item output will be at the right end of the space specified by `width`. In other words, any extra blanks are placed *before* the item output. (Setting this flag automatically unsets the flag `ios::left`.)	Set
`ios::left`	If this flag is set and some field-width value is given with a call to the member function `width`, then the next item output will be at the left end of the space specified by `width`. In other words, any extra blanks are placed *after* the item output. (Setting this flag automatically unsets the flag `ios::right`.)	Not set

Форматиращи флагове за setf

- Форматираща функция е width. Тя задава ширината на полето в брой позиции, в които да се разположи съответната данна.

```
cout << "Start Now";
cout.width(4);
cout << 7 << endl;
```

```
Start Now   7
```

- Действието на този флаг може да се замени от манипулатора setw(число)
-

```
cout << "Start" << setw(4) << 10
     << setw(4) << 20 << setw(6) << 30;
```

- Функцията unsetf отменя действието на поставените флагове

```
cout.unsetf(ios::showpos);
```

- В този пример се преустановява изписването на знаци на числото

```
cout.setf(ios::fixed);
cout.setf(ios::showpoint);
cout << "$" << setprecision(2) << 10.3 << endl
     << "$" << 20.5 << endl;
```

```
$10.30
$20.50
```

Форматиране на числа с десетична точка

- Възможно е в някои компилатори горният пример да не използва флагове (cout.setf), а изходният манипулатор setprecision (цяло число). Разликата тук е че цялото число включва и позицията на десетичната запетая
- Примерът е следния:

cout<< endl<<"The price is $ "<<setprecision(3)<<price;

```
The price is $78.50
```

- Отмяната на setprecision се извършва с setprecision(0)

Въвеждане на данни чрез cin

- cin е обект, който е свързан със стандартното входно устройство
- Операция >> (пренасочване), пренасочва данните „отвън към компютъра“

```
cout << "Enter the number of candy bars in a package\n";
cout << "and the weight in ounces of one candy bar.\n";
cout << "Then press return.\n";
cin >> number_of_bars;
cin >> one_weight;
```

```
cout << "Enter the number of candy bars in a package\n"
cout << "and the weight in ounces of one candy bar.\n"
cout << "Then press return.\n";
cin >> number_of_bars >> one_weight;
```

- Чрез cin може да се въвеждат повече от една данна
- Числата трябва да се въвеждат, разделени с един или повече празни интервали или с край на реда.
- Примерно числата 12 и 5 трябва да се въведат 12 5. Ако се въведат без интервал компютърът отчита едно число 125.
- Ако данните не се въведат, компютърът продължава да ги изисква до крайното им въвеждане

Форматиран вход с scanf

- Форматиран вход за всички типове данни
- Въвеждане на специфични символи от входния поток
- Пропускане на специфични символи от входния поток
- Пример:

```
scanf("%e%f%g", &a, &b, &c);
```

Conversion specifier	Description
Integers	
d	Read an optionally signed decimal integer. The corresponding argument is a pointer to integer.
i	Read an optionally signed decimal, octal, or hexadecimal integer. The corresponding argument is a pointer to integer.
o	Read an octal integer. The corresponding argument is a pointer to unsigned integer.
u	Read an unsigned decimal integer. The corresponding argument is a pointer to unsigned integer.
x or X	Read a hexadecimal integer. The corresponding argument is a pointer to unsigned integer.
h or l	Place before any of the integer conversion specifiers to indicate that a short or long integer is to be input.
Floating-point numbers	
e, E, f, g or G	Read a floating-point value. The corresponding argument is a pointer to a floating-point variable
l or L	Place before any of the floating-point conversion specifiers to indicate that a double or long double value is to be input.
Characters and strings	
c	Read a character. The corresponding argument is a pointer to char, no null ('\0') is added.
s	Read a string. The corresponding argument is a pointer to an array of type char that is large enough to hold the string and a terminating null ('\0') character.

Символен вход/изход

- Функция get четене на символ от входния поток от тип char (буфериран вход)

```
char next_symbol;
cin.get(next_symbol);
```

- Символът '\n' се използва и за вход

```
cout << "Enter a line of input and I will echo it:\n";
char symbol;
do
{
    cin.get(symbol);
    cout << symbol;
} while (symbol != '\n');
cout << "That's all for this demonstration.";
```

```
Enter a line of input and I will echo it:
Do Be Do 1 2    34
Do Be Do 1 2    34
That's all for this demonstration.
```

- Функцията put извежда символ към изходния поток `cout.put('a');`
- Двете функции ползват iostream.h

Символен вход/изход

- Функция getch()- четене на символ от входен поток без exo
- Функция getche() – четене на символ от входен поток с exo

```
char symbol=getch();
putch(symbol);
char symbol=getche();  //   не се изисква putch()
```

- Функциите getch(), getche(), putch(), putche() изискват conio.h
- Функциите getchar() и putchar() изискват stdio.h
- Функцията getchar() използва линейно буфериране. Изисква натискане на Enter

```
char symbol;
symbol=getchar();
```

Някои функции за работа със символи

- Тези функции изискват заглавния файл ctype.h

Function	Description	Example
toupper(*Char_Exp*)	Returns the uppercase version of *Char_Exp*.	*char* c = toupper('a'); cout << c; **Outputs:** A
tolower(*Char_Exp*)	Returns the lowercase version of *Char_Exp*.	*char* c = tolower('A'); cout << c; **Outputs:** a

isupper(*Char_Exp*)	Returns *true* provided *Char_Exp* is an uppercase letter; otherwise, returns *false*.	*if* (isupper(c))
		  cout << c << " is uppercase.";
		else
		  cout << c
		  << " is not uppercase.";

islower(*Char_Exp*)	Returns *true* provided *Char_Exp* is a lowercase letter; otherwise, returns *false*.	*char* c = 'a'; *if* (islower(c)) cout << c << " is lowercase."; **Outputs:** a is lowercase.

isalpha(*Char_Exp*)	Returns *true* provided *Char_Exp* is a letter of the alphabet; otherwise, returns *false*.	*char* c = '$';
		if (isalpha(c))
		  cout << c << " is a letter.";
		else
		  cout << c
		  << " is not a letter.";
		Outputs: $ is not a letter.

isdigit(*Char_Exp*)	Returns *true* provided *Char_Exp* is one of the digits '0' through '9'; otherwise, returns *false*.	*if* (isdigit('3'))
		  cout << "It's a digit.";
		else
		  cout << "It's not a digit.";
		Outputs: It's a digit.

isspace(*Char_Exp*)	Returns *true* provided *Char_Exp* is a whitespace character, such as the blank or new-line symbol; otherwise, returns *false*.	*//Skips over one "word" and*
		//sets c equal to the first
		//whitespace character after
		//the "word":
		do
		{
		  cin.get(c);
		} *while* (! isspace(c));

Стрингов вход и изход

- Въвеждане на стринг чрез cin>> - въвеждането спира при първия срещнат празен интервал
- cin.getline(име_на_стринга,максим_брой_ символи) – въвеждат се максим. брой символи за стринга или до достигане края на реда
- При въвеждане на стринга ВАРНА10, функцията cin.getline(city,6) въвежда за променливата city стойност Варна (1 символ за края на стринга \0)
- cin.getline() изисква iostream.h
- Възможно е ползването на gets() за въвеждане и функциите cout<< или puts() – за извеждане на низове
- Използвайте функцията cin.ignore() или други подобни функции за изчистване на входния буфер.