NHẬP MÔN LẬP TRÌNH

CHƯƠNG 3.3: NHẬP/ XUẤT CƠ BẢN

BASIC INPUT/ OUTPUT

ThS. Nguyễn Thị Ngọc Diễm
diemntn@uit.edu.vn

Nội dung

1. Câu lệnh xuất dữ liệu
2. Câu lệnh xuất std::cout<<
3. Câu lệnh xuất printf
4. Câu lệnh nhập
5. Câu lệnh std::cin>>, nhập ký tự: cin và char, nhập chuỗi: cin và strings
6. Câu lệnh scanf
7. Xóa bộ nhớ đệm stdin

1. Câu lệnh xuất

Có 2 cách:

Cách 1: Sử dụng lệnh xuất trong C++: **cout**

Cách 2: Sử dụng lệnh xuất trong C: **printf**

Lựa chọn tùy thuộc vào lập trình viên.

2. Câu lệnh xuất cout (C++)

Thư viện:

```
#include <iostream>
```

Cú pháp:

```
std::cout << Tham_số_1 << Tham_số_2 << ... << Tham_số_k;
```

Tham số có thể:

- Văn bản thường (literal text)
- Ký tự điều khiển (escape sequence)
- Biến, hằng số, biểu thức, hàm

Ví dụ:

```
int i=0;
std::cout<<"Chuong trinh xuat gia tri i \n";
std::cout<<"gia tri la "<<i;
```

```
Chuong trinh xuat gia tri i
10
```

2. Câu lệnh xuất cout (C++)

Cú pháp:

Ví dụ:

```
std::cout<<"Chuoi can in";
```

```
// Thư vien ho tro nhap xuat
#include <iostream>
int main() {
          // Xuat thong bao
          std::cout<<“Chao ban! Toi co the giup gi
khong?";
          // Cau lenh dung man hinh kiem tra ket qua
          std::cin.get();
          return 0;
}
```

Chao ban! Toi co the giup gi khong?

2. Câu lệnh xuất cout (C++)

- Ký tự điều khiển (escape sequence)
  - Gồm dấu \ và một ký tự như trong bảng sau:

Escape sequence	Description	Escape sequence	Description
\'	single quote	**\r**	carriage return
\"	double quote	**\t**	horizontal tab
\?	question mark	**\v**	vertical tab
****	backslash	**\nnn**	arbitrary octal value
\a	audible bell	**\xnn**	arbitrary hexadecimal value
\b	backspace	**\unnnn**	universal character name (arbitrary Unicode value); may result in several characters
\f	form feed - new page	**\v**	vertical tab
\n	line feed - new line	**\nnn**	arbitrary octal value

2. Câu lệnh xuất cout (C++)

```
#include <iostream>
int main() {
          std::cout<<"Chao ban!\nToi co the giup gi khong?";
          std::cin.get();
          return 0;
}
```

Chao ban!
Toi co the giup gi khong?

Ví dụ

Dùng cout hiển thị các chuỗi sau lên màn hình:

1. **Toi dang hoc mon NNMLT**
2. **One two three**
   Four five six
3. **Day la lop "Nhap mon lap trinh" phai khong?**

```
cout << "Toi dang hoc mon NNMLT";
```

```
cout << “One \t two \t three \nFour \t five \t six";
```

```
cout << "Day la lop \"Nhap mon lap trinh\" phai khong\?";
```

2. Một số thiết lập của cout khi xuất

Thiết lập độ rộng

Cú pháp: **cout.width(n)** - Với n là độ rộng mới

Chú ý: Độ rộng quy định n chỉ có tác dụng cho một giá trị xuất.
Sau đó C++ lại áp dụng độ rộng quy định bằng 0.

Ví dụ:

```
#include <iostream>
int main() {
     int a = 1706;
     std::cout << a << "\n";
     std::cout.width(10);
     std::cout << a;
     std::cin.get();
    return 0;
}
```

2. Một số thiết lập của cout khi xuất

Độ chính xác số thực:

Cú pháp: **cout.precision(n)** - Với n là độ chính xác áp dụng

Chú ý: độ chính xác được thiết lập sẽ có hiệu lực cho tới khi gặp một câu lệnh thiết lập độ chính xác mới

Ví dụ:

```
#include <iostream>
int main() {
     float x = 176.859;
     std::cout << x << "\n";
     std::cout.precision(5);
     std::cout << x << "\n";
     std::cin.get();
    return 0;
}
```

2. Một số thiết lập của cout khi xuất

Other Format flag manipulators:

- Thư viên <iomanip>, namespace std:
  - setw
  - setfill
  - setprecision
- Thư viện <ios>, namespace std:
  - setprecision,
  - hex, dec, oct
  - showbase, noshowbase
  - uppercase, nouppercase
  - showpos, noshowpos
  - fixed
  - scientific
  - …

3. Câu lệnh xuất printf (ngôn ngữ C)

Thư viện

#include <stdio.h> (standard input/output)

Cú pháp

printf(<chuỗi định dạng>[, <đs1>, <đs2>, …]);

<chuỗi định dạng> là cách trình bày thông tin xuất và được đặt trong cặp nháy kép " ", gồm:

- Văn bản thường (literal text)
- Ký tự điều khiển (escape sequence)
- Đặc tả (conversion specifier)

3. Đặc tả sử dụng hàm printf (Specifier)

A *format specifier* follows this prototype:

%[flags][width][.precision][length]specifier

Specifier

Specifier	Output	Example
d *or* i	Signed decimal integer	392, -392
u	Unsigned decimal integer	7235
o	Unsigned octal	610
x	Unsigned hexadecimal integer	7fa
X	Unsigned hexadecimal integer (uppercase)	7FA
f	Decimal floating point, lowercase	392.65
F	Decimal floating point, uppercase	392.65
e	Scientific notation (mantissa/exponent), lowercase	3.9265e+2
E	Scientific notation (mantissa/exponent), uppercase	3.9265E+2
g	Use the shortest representation: %e or %f	392.65

Specifier (tt)

Specifier	Output	Example
G	Use the shortest representation: %E or %F	392.65
a	Hexadecimal floating point, lowercase	-0xc.90fep-2
A	Hexadecimal floating point, uppercase	-0XC.90FEP-2
c	Character	a
s	String of characters	sample
p	Pointer address	b8000000
n	Nothing printed. The corresponding argument must be a pointer to a signed int. The number of characters written so far is stored in the pointed location.	
%	A % followed by another % character will write a single % to the stream.	%

flags

flags	description
-	Left-justify within the given field width; Right justification is the default (see *width* sub-specifier).
+	Forces to preceed the result with a plus or minus sign (+ or -) even for positive numbers. By default, only negative numbers are preceded with a - sign.
(space)	If no sign is going to be written, a blank space is inserted before the value.
#	Used with o, x or X specifiers the value is preceeded with 0, 0x or 0X respectively for values different than zero. Used with a, A, e, E, f, F, g or G it forces the written output to contain a decimal point even if no more digits follow. By default, if no digits follow, no decimal point is written.
0	Left-pads the number with zeroes (0) instead of spaces when padding is specified (see *width* sub-specifier).

width

width	description
(number)	Minimum number of characters to be printed. If the value to be printed is shorter than this number, the result is padded with blank spaces. The value is not truncated even if the result is larger.
*	The *width* is not specified in the *format* string, but as an additional integer value argument preceding the argument that has to be formatted.

precision

.precision	description
.number	For integer specifiers (d, i, o, u, x, X): *precision* specifies the minimum number of digits to be written. If the value to be written is shorter than this number, the result is padded with leading zeros. The value is not truncated even if the result is longer. A *precision* of 0 means that no character is written for the value 0. For a, A, e, E, f and F specifiers: this is the number of digits to be printed **after** the decimal point (by default, this is 6). For g and G specifiers: This is the maximum number of significant digits to be printed. For s: this is the maximum number of characters to be printed. By default all characters are printed until the ending null character is encountered. If the period is specified without an explicit value for *precision*, 0 is assumed.
.*	The *precision* is not specified in the *format* string, but as an additional integer value argument preceding the argument that has to be formatted.

Ví dụ

- int a = 10, b = 20;
- printf(“%d”, a); ➔ Xuất ra 10
- printf(“%d”, b); ➔ Xuất ra 20
- printf(“%d %d”, a, b); ➔ Xuất ra 10 20

- float x = 15.06;
- printf(“%f”, x); ➔ Xuất ra 15.060000
- printf(“%f”, 1.0/3); ➔ Xuất ra 0.333333

Ví dụ

```
int a = 1706;
float x = 176.85;
printf(“%10d”, a);printf(“\n”);
printf(“%2f”, x);printf(“\n”);
printf(“%.2f”, x);printf(“\n”);
```

Phối hợp các thành phần

- int a = 1, b = 2;
- Xuất 1 cong 2 bang 3 và xuống dòng.
  - printf("%d", a); // Xuất giá trị của biến a
  - printf(" cong "); // Xuất chuỗi " cong "
  - printf("%d", b); // Xuất giá trị của biến b
  - printf(" bang "); // Xuất chuỗi " bang "
  - printf("%d", a + b); // Xuất giá trị của a + b
  - printf("\n"); // Xuất điều khiển xuống dòng \n

  ➔ printf("%d cong %d bang %d\n", a, b, a+b);

Ví dụ:

```c
/* printf example */
#include <stdio.h>
int main() {
    printf ("Characters: %c %c \n", 'a', 65);
    printf ("Decimals: %d %ld\n", 1977, 650000L);
    printf ("Preceding with blanks: %10d \n", 1977);
    printf ("Preceding with zeros: %010d \n", 1977);
    printf ("Some different radices: %d %x %o %#x %#o \n", 100,
    100, 100, 100, 100);
    printf ("floats: %4.2f %+.0e %E \n", 3.1416, 3.1416,
    3.1416);
    printf ("Width trick: %*d \n", 5, 10);
    printf ("%s \n", "A string");
    return 0;
}
```

? Dùng cout để hiển thị thay printf nội dung trên.

4. Câu lệnh nhập

Có 2 cách:

Cách 1: Sử dụng lệnh nhập trong C++: cin

Cách 2: Sử dụng nhập trong C: scanf

Lựa chọn tùy thuộc vào lập trình viên

5. Câu lệnh nhập std::cin>> (C++)

Thư viện:

```
#include <iostream>
```

Cú pháp:

```
std::cin>>Tham_số_1>>Tham_số_2>>...>>Tham_số_k;
```

Lưu ý: Tham số ko có dạng chuỗi

Ví dụ:

```
#include <iostream>
int main() {
        int namsinh = 0; //Buoc 1
        std::cout<<"Nam sinh: "; // Buoc 2
        std::cin>>namsinh; // Buoc 3;
        std::cin.get();
        return 0;
}
```

Ví dụ

```
#include <iostream>
int main() {
        int a = 0, b = 0;
        std::cout<<"Chuong trinh cong 2 so a, b"<<"\n";
        std::cout<<"a: ";
        std::cin>>a;
        std::cout<<"b: ";
        std::cin>>b;
        std::cout<<"a + b = "<<a + b<<"\n";
        std::cin.get();
        return 0;
}
```

Chuong trinh cong 2 so a,b
a = 5
b = 6
a + b = 11

Viết chương trình tính a + b, a – b, a * b, a / b.

Nhập ký tự: cin and char

Có thể dung std::cin hoặc hàm std::cin.get() để lấy 1 ký tự từ dòng nhập stream.

Ví dụ:

`char mychar;` `mychar = std::cin.get();`	`char mychar;` `std::cin >> mychar;`

```
#include <iostream> // std::cin, std::cout
int main () {
  char first, last;
  std::cout << "surname: ";
  first = std::cin.get(); // get one character
  std::cout << "lastname: ";
  std::cin >> last; // get one character
  std::cout << std::endl << "Your initials: " << first << last
  << '\n';
  return 0;
}
```

Nhập chuỗi: cin and strings

- The extraction operator can be used on cin to get strings of characters in the same way as with fundamental data types:

```
std::string mystring;
Std::cin >> mystring;
```

- However, **cin** extraction always considers spaces (whitespaces, tabs, new-line...) as terminating the value being extracted, and thus extracting a string means to always extract a single word, not a phrase or an entire sentence.

=> Dùng hàm std::getline

```
std::string mystring;
std::getline (std::cin, mystring);
```

Nhập chuỗi: cin and strings

```
// cin with strings
#include <iostream>
#include <string>

int main () {
    std::string mystr;
    std::cout << "What's your name? ";
    std::getline (std::cin, mystr);
    std::cout << "Hello " << mystr << ".\n";
    std::cout << "What is your favorite team? ";
    std::getline (std::cin, mystr);
    std::cout << "I like " << mystr << " too!\n";
    return 0;
}
```

6. Câu lệnh nhập scanf (Ngôn ngữ C)

Thư viện

- #include <stdio.h> (standard input/output)

Cú pháp

- scanf(<chuỗi định dạng>[, <đs1>, <đs1>, …]);
- <chuỗi định dạng> giống định dạng xuất nhưng chỉ có các đặc tả
- Các đối số là tên các biến sẽ chứa giá trị nhập và được đặt trước dấu &

6. Câu lệnh nhập scanf

Ví dụ, cho a và b kiểu số nguyên

- scanf(“%d”, &a); // Nhập giá trị cho biến a
- scanf(“%d”, &b); // Nhập giá trị cho biến b
- ➔ scanf(“%d%d”, &a, &b);
- Các câu lệnh sau đây sai
  - scanf(“%d”, a);// Thiếu dấu &
  - scanf(“%d”, &a, &b);// Thiếu %d cho biến b
  - scanf(“%f”, &a); // a là biến kiểu số nguyên
  - scanf(“%9d”, &a); // không được định dạng
  - scanf(“a = %d, b = %d”, &a, &b”);

7. Xóa bộ nhớ đệm

- Mọi ký tự được gõ vào từ bàn phím đều được đưa vào bộ nhớ đệm trước khi được gán vào biến. Nếu nhập số/ký tự và phím Enter để kết thực việc nhập bằng scanf hoặc cin, biến chỉ nhận số chứ không nhận được ký tự Enter, vì vây ký tự Enter vẫn còn trong bộ nhớ đệm. Đến khi nhập chuỗi, các hàm nhập này nhận được ký tự Enter thì dừng nhập luôn và chương trình vẫn chạy tiếp. Điều này khiến kết quả không giống như mong đợi.
- Vì vậy trước khi nhập một chuỗi trong màn hình console, ta cần có thao tác xóa bộ nhớ đệm bàn phím stdin. Nếu không có thể kết quả nhập chuỗi bị sai hoặc bị trôi đi mất.

Xóa bộ nhớ đệm

- Các hàm sau dung để xóa bộ nhớ đệm:

Hàm	Thư viện	Ý nghĩa
fflush(stdin)	<stdio.h>	Dùng để xóa bộ nhớ đệm cho stream nhập từ bàn phím.
std::cin.ignore(streamsize n = 1, int delim = EOF))	<iostream>	Tách và bỏ qua n các ký tự trong bộ nhớ đệm stdin. Bỏ qua hay loại bỏ một số lượng ký tự n trong bộ đệm stdin, số lượng ký tự còn lại (nếu có) vẫn ở trên stdin.hoặc bỏ qua đến khi gặp ký tự *delim*.
std::cin.ignore()	<iostream>	Mặc định là bỏ 1 ký tự trong bộ đệm.

Ví dụ 1:

```
// cin with number and string
#include <iostream> // std::cout,
std::cin, std::getline
#include <string> // std::string
#include <stdio.h> // fflush(stdin)


int main () {
	std::string name;
	int age;

	std::cout << "Age: ";
	std::cin >> age;

	std::cin.ignore(6, '\n');
	std::cout << "Name: ";
	std::getline(std::cin, name);

	std::cout << std::endl << std::endl
	<< "Output: " << name << ", " <<
	age;
	return 0;
}
```

Chạy lần 1:

Age: 30 123
// std::cin.ignore(6, '\n'): gặp enter nên kết thực việc nhập, dãy " 123" gồm 4 ký tự bị bỏ qua, stdin rỗng
Name: ngoc diem

Output: ngoc diem, 30

Chạy lần 2:

Age: 30 123456789
// std::cin.ignore(6, '\n'): gặp enter nên kết thực việc nhập, dãy " 12345" gồm 6 ký tự bị bỏ qua, stdin còn lại là dãy "6789"
Name:

Output: 6789, 30

Ví dụ 2:

```
#include <iostream> // std::cin, std::cout
int main () {
      char first, last;
      std::cout << "surname: ";
      first = std::cin.get(); // get one
      character
      std::cin.ignore(6, '\n'); // ignore
      until enter
      std::cout << "lastname: ";
      std::cin >> last; // get one
      character
      std::cout << std::endl << "Your
      initials: " << first << last << '\n';
      return 0;
}
```

Chạy lần 1:

surname: d123456789
// std::cin.ignore(6, '\n'): gặp enter nên kết thực việc nhập, dãy " 123456" gồm 6 ký tự bị bỏ qua, stdin còn lại là dãy "789"
lastname:

Your initials: d7

Chạy lần 2:

surname: d123
// std::cin.ignore(6, '\n'): gặp enter nên kết thực việc nhập, dãy " 123" gồm 3 ký tự bị bỏ qua, stdin rỗng
lastname: 456
//gặp enter nên kết thực việc nhập, stdin còn lại là dãy "56"

Your initials: d4

Ví dụ

- Viết chương trình Nhập, xuất thông tin của 1 sinh viên. Biết thông tin của sinh viên bao gồm: { Họ tên, Giới tính, Điểm toán, Điểm tin, Điểm trung bình, Xếp loại, Kết quả}

Biến	Kiểu dữ liệu	Ví dụ
Họ tên sinh viên	(chuỗi)	"Nguyen Xuan Minh"
Giới tính	(char)	M (Male)/ F (Female)
Điểm Toán	(số nguyên)	8
Điểm Tin	(số nguyên)	9
Điểm Trung bình	float	8.5
~~Xếp loại~~	~~Chuỗi~~	~~Giỏi~~
~~Kết quả~~	~~bool~~	~~1 (Đậu)/ 0 (Rớt)~~

9. Một số hàm hữu ích khác

Thư viện	Ví dụ	Ý nghĩa
<iostream>	std::boolalpha std::noboolalpha	Nhập và xuất dạng chuỗi true/false. Vị trí: đặt trước các dòng lệnh cout. Có tác dụng cho tất cả các dòng lệnh cout bên dưới. (*)

	std::cin.setf(std::ios::boolalpha);
	std::cout.setf(std::ios::boolalpha);
	std::cout.unsetf(std::ios::boolalpha);
	std::cin.unsetf(std::ios::boolalpha);
		Nhập và xuất dạng chuỗi true/false.
		Vị trí: đặt trước các dòng lệnh cin, cout. Có tác dụng cho tất cả các dòng lệnh cin, cout bên dưới.

	std::hex, std::oct, std::scientific	Vị trí: (*)
	std::showbase, std::noshowbase	Vị trí: như (*)
	std::nouppcase, std::uppercase	Vị trí: như (*)
	std::fixed	Vị trí: như (*)
	std::internal, std:left, std::right	Vị trí: đặt trước các dòng lệnh cout, tác dụng cho dòng 1 stream sau.
<iomanip>	std::setw	Vị trí: đặt trước các dòng lệnh cout, tác dụng cho dòng 1 stream sau.
	std::setprecision	Vị trí: như (*)
	std::setfill	Vị trí: như (*)
	std::setbase	Vị trí: như (*)
	std::setiosflags, std::resetiosflags	Vị trí: như (*)

Bài tập minh họa

1. Nhập năm sinh của một người và tính tuổi của người đó.
2. Nhập 2 số a và b. Tính tổng, hiệu, tính và thương của hai số đó.
3. Nhập tên sản phẩm, số lượng và đơn giá. Tính tiền và thuế giá trị gia tăng phải trả, biết:
   a. tiền = số lượng * đơn giá
   b. thuế giá trị gia tăng = 10% tiền
4. Nhập bán kính của đường tròn. Tính chu vi và diện tích của hình tròn đó.

Bài tập minh họa

1. Nhập năm sinh của một người và tính tuổi của người đó.

```
#include <iostream>
int main() {
    int namsinh = 0;
    std::cout<<"Vui long nhap nam sinh: ";
    std::cin>>namsinh;
    std::cout<<"Ban "<<2016-namsinh<<" tuoi"<<"\n";
    std::cin.get();
    return 0;
}
```

Bài tập minh họa

2. Nhập 2 số a và b. Tính tổng, hiệu, tính và thương của hai số đó.

```
#include <iostream>
int main() {
    int a, b;
    std::cout<<"Nhap a = ";
    std::cin>>a;
    std::cout<<"Nhap b = ";
    std::cin>>b;
    std::cout<<"a + b = "<<a+b<<"\n";
    std::cout<<"a - b = "<<a-b<<"\n";
    std::cout<<"a * b = "<<(long long)a*b<<"\n";
    std::cout<<"a / b = "<<(double)a/b<<"\n";
    std::cin.get();
    return 0;
}
```

Bài tập minh họa

3. Nhập tên sản phẩm, số lượng và đơn giá. Tính tiền và thuế giá trị gia tăng phải trả, biết:
   a. tiền = số lượng * đơn giá
   b. thuế giá trị gia tăng = 10% tiền

```
#include <iostream>
int main() {
    int so_luong = 0, don_gia = 0;
    std::cout<<"Vui long nhap so luong: ";
    std::cin>>so_luong;
    std::cout<<"Vui long nhap don gia: ";
    std::cin>>don_gia;
    std::cout<<"Tien: "<<so_luong * don_gia<<"\n";
    std::cout<<"VAT: "<<so_luong * don_gia * 0.1<<"\n";
    std::cin.get();
    return 0;
}
```

Bài tập minh họa

4. Nhập bán kính của đường tròn. Tính chu vi và diện tích của hình tròn đó.

```
#include <iostream>
#define PI 3.14
int main() {
    float r = 0;
    std::cout<<"Nhap ban kinh duong tron: ";
    std::cin>>r;
    std::cout<<"Chu vi: "<<2 * PI * r<<"\n";
    std::cout<<"Dien tich: "<<PI * r * r<<"\n";
    return 0;
}
```

Bài tập về nhà

1. Cho số xe (gồm 4 chữ số) của bạn. Cho biết số xe của bạn được mấy nút?
2. Cho 1 ký tự chữ thường. In ra ký tự chữ hoa tương ứng.
3. Cho 3 số nguyên. Cho biết số lớn nhất và nhỏ nhất?
4. Viết chương trình cho 2 giờ (giờ, phút, giây) và thực hiện cộng, trừ 2 giờ này.
5. Tổng các bội số của 3 và 5 nhỏ hơn 10 là 23.
   Ví dụ: Ta có các bội số: 3, 5, 6, 9 → Tổng: 23
   Tính tổng các bội số của 3 và 5 nhỏ hơn 1000.

Chúc các em học tốt!