LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG
(Object Oriented Programming)

KHOA CÔNG NGHỆ THÔNG TIN, ĐH SƯ PHẠM HÀ NỘI

HÀ NỘI, THÁNG 9-2022

PHẦN 1: LẬP TRÌNH CƠ BẢN

BÀI 1:
BIẾN, KIỂU DỮ LIỆU,
VÀ CÁC CẤU TRÚC CƠ BẢN

NỘI DUNG CHÍNH

- Ôn tập lại các kiến thức cơ bản về lập trình
- Hiểu và sử dụng được các biến, kiểu dữ liệu cơ bản
- Hiểu và sử dụng được các cấu trúc tuần tự, cấu trúc rẽ nhánh, cấu trúc lặp
- Viết được các chương trình giải các bài toán đơn giản

BIẾN VÀ KIỂU DỮ LIỆU

- Biến
- Kiểu dữ liệu
- Ép kiểu
- Toán tử và biểu thức

BIẾN VÀ CÁCH ĐẶT TÊN BIẾN

- **Biến** là một vùng nhớ được đặt tên
- Mỗi biến có một kiểu dữ liệu
- Kích thước của biến phụ thuộc vào kiểu dữ liệu của biến
- Tên biến được đặt theo quy tắc và gợi nhớ
  - Bắt đầu bằng chữ cái (a..z, A..Z, _)
  - Tiếp theo là chữ cái hoặc chữ số (a..z, A..Z, _, 0..9)
  - Không trùng với từ khoá

Chú ý: C++ phân biệt chữ HOA và chữ thường.

DANH SÁCH TỪ KHÓA TRONG C++

Từ khóa: Là từ những từ được ngôn ngữ lập trình sử dụng cho một mục đích riêng.

Trong chương trình, không được đặt tên biến trùng với tên của từ khóa.

alignas	decltype	namespace	struct
alignof	default	new	switch
and	delete	noexcept	template
and_eq	double	not	this
asm	do	not_eq	thread_local
auto	dynamic_cast	nullptr	throw
bitand	else	operator	true
bitor	enum	or	try
bool	explicit	or_eq	typedef
break	export	private	typeid
case	extern	protected	typename
catch	false	public	union
char	float	register	unsigned
char16_t	for	reinterpret_cast	using
char32_t	friend	return	virtual
class	goto	short	void
compl	if	signed	volatile
const	inline	sizeof	wchar_t
constexpr	int	static	while
const_cast	long	static_assert	xor
continue	mutable	static_cast	xor_eq

VÍ DỤ CÁCH ĐẶT TÊN BIẾN

Tên	Đúng/Sai
Birthday	Đ.
Too_hot?	S. (Không được có dấu "?")
First_Initial	Đ.
1stprogram	S. (Không được bắt đầu bằng số)
down.to.earth	S. (Không được có dấu ".")
see you	S. (Không được có dấu cách)
OldName	Đ. (Tốt, nên viết hoa đầu từ tiếng Anh)
case	S. (Trùng từ khoá)
One+Two	S. (Không được có dấu "+")

DỮ LIỆU VÀ KIỂU DỮ LIỆU

- **Dữ liệu**
  - Là tất cả những gì được máy tính lưu trữ và xử lý
  - Tồn tại dưới nhiều dạng khác nhau, không nhất thiết phải là số
  - Thể hiện các đối tượng cụ thể cần xử lý (giá tiền, tên, tuổi, văn bản, tín hiệu...)
- **Kiểu dữ liệu**
  - Một tập hợp các giá trị mà một biến thuộc kiểu đó có thể nhận
  - Trên tập hợp đó xác định một số phép toán.
    - +, -, *, / với kiểu số thực
    - /, % với kiểu số nguyên
    - &&, ||, !, ^^ với kiểu logic mệnh đề

CÁC KIỂU DỮ LIỆU CHUẨN

- **Các kiểu ký tự**: Biểu diễn một ký tự trong một bảng mã nào đó, chẳng hạn 'A' hay '$'... Phổ biến nhất là kiểu char, biểu diễn các ký tự trong bảng mã ASCII
- **Các kiểu số nguyên**: Biểu diễn một số nguyên, các kiểu số nguyên được phân loại bởi kích thước và miền giá trị (có dấu hay không dấu)
- **Các kiểu số thực**: Biểu diễn một số thực, các kiểu số thực được phân loại bởi kích thước và độ chính xác
- **Các kiểu logic:** Chỉ nhận một trong hai giá trị Đúng (true) hoặc Sai (false)

CÁC KIỂU SỐ NGUYÊN

Kiểu dữ liêu	Kích thước	Miền giá trị
char	1 byte	[-127, 127]
short (short int)	2 byte	[-32768, 32767]
int	4 byte	[-2147483648, 2147483647]
long (long int)	4 byte	[-2147483 648, 2147483647]
long long (long long int)	8 byte	[-92233720368547758 08, 9223372036854775807]

Các kiểu số nguyên có dấu

Các kiểu số nguyên không dấu

Kiểu dữ liêu	Kích thước	Miền giá trị
unsigned char	1 byte	[0, 255]
unsigned short	2 byte	[0, 65 535]
unsigned int	4 byte	[0, 4 294 967 295]
unsigned long	4 byte	[0, 4 294 967 295]
unsigned long long	8 byte	[0, 18 446 744 073 709 551 615]

CÁC KIỂU SỐ THỰC

Kiểu dữ liêu	Kích thước	Miền giá trị	Độ chính xác sau phần thập phân
`float`	4 byte	$[1.17549\times10^{-38}, 1.17549\times10^{+38}]$	6 chữ số
`double`	8 byte	$[2.22507\times10^{-308}, 1.79769\times10^{+308}]$	15 chữ số
`long double`	8 byte	$[2.22507\times10^{-308}, 1.79769\times10^{+308}]$	15 chữ số

KIỂU bool VÀ KIỂU void

Kiểu logic

- Dữ liệu kiểu logic chỉ là một ô nhớ lưu trữ hai giá trị ứng với hai trạng thái đúng (true) hoặc sai (false)
- Kiểu logic trong C++ là bool, chiếm 8 bit (1 byte)

Kiểu void

- void: Không là kiểu gì cả
- void là một kiểu dữ liệu đặc biệt trong C++, có thể coi void là kiểu mà người ta không quan tâm tới giá trị lưu trong kiểu đó.

KHAI BÁO BIẾN

- **Biến**
  - Nhận giá trị thuộc một kiểu dữ liệu nào đó, chiếm lượng bộ nhớ do kiểu dữ liệu quy định
  - Có thể thay đổi giá trị trong chương trình
- **Mỗi khai báo có dạng**: T v1, v2, …, vn;
- Trong đó
  - T: Tên kiểu dữ liệu
  - v1, v2, …, vn: Các tên biến cách nhau bởi dấu phẩy ","
- Khai báo biến trong C++ có thể nằm ở bất cứ đâu trong chương trình
- Đôi khi người ta muốn một biến khi khai báo nhận luôn một giá trị khởi tạo. Khi đó có thể đưa giá trị khởi tạo vào trong khai báo biến

```
int i, j, k;
double x, y, z;
char ch;
bool flag;
```

XUẤT SỐ THỰC THEO KHUÔN DẠNG

- Các kiểu dữ liệu số thực có khuôn dạng mặc định khi in ra màn hình
- Muốn thay đổi khuôn dạng in với k chữ số sau dấu chấm thập phân
  - Khai báo sử dụng thư viện iomanip
  - Đẩy ra luồng ra chuẩn hai giá trị std::fixed và std::setprecision(k)

(có thể viết fixed, setprecision(k) nếu đã khai báo using namespace std;)

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

int main() {
  double d = 12.345678;
  cout << fixed << setprecision(2);
  cout << d; //12.35
  return 0;
}
```

THƯ VIỆN <bits/stdc++.h>

- C++ đi kèm với rất nhiều thư viện hỗ trợ
- Việc #include<từng thư viện> khá dài dòng và mất thời gian
- Một header file chung, chỉ cần đưa khai báo

  #include<bits/stdc++.h>
- Để sử dụng tất cả thư viện chuẩn
- Như vậy sẽ giúp cho việc khai báo các thư viện được nhanh chóng, tuy nhiên, việc #include<bits/stdc++.h> làm kéo dài thời gian dịch chương trình và tăng kích thước mã máy.
- Nếu chương trình nhỏ, dùng ít thư viện thì không nên sử dụng cách này.

TOÁN TỬ VÀ BIỂU THỨC

- Biểu thức bao gồm các toán tử và các toán hạng. Chẳng hạn biểu thức $a + b$ có toán tử là phép cộng "+" và hai toán hạng là a và b.
- Trong C++, toán hạng có thể là hằng, biến, hàm, hoặc một biểu thức khác. Những biểu thức này có định kiểu và toán tử phải tương thích với kiểu của toán hạng.

BIỂU THỨC GÁN (CÂU LỆNH GÁN)

- Biểu thức gán (assignment) có cú pháp: «Tên biến» = «Biểu thức»;
- Ở đây «Tên biến» là tên một biến đã khai báo, dấu "=" là ký hiệu toán tử gán, «Biểu thức» là giá trị gán cho biến có tên ở vế trái.
- Khi gặp toán tử gán, máy sẽ tính giá trị của «Biểu thức» ở vế phải, lấy giá trị đó gán cho biến với «Tên biến» chỉ ra trong vế trái. Giá trị cũ của biến bị hủy, thay bằng giá trị mới.
- Biểu thức gán cũng được coi là một câu lệnh trong C++, gọi là lệnh gán

Phân tích ví dụ

```
1 | int x, y;
2 | x = 4;
3 | x = x + 8; //x = 4 + 8 = 12
4 | x = x * 2; //x = 12 * 2 = 24
5 | y = (x = x * 5) + 3; //x = 24 * 5 = 120; y = 120 + 3 = 123
```

CÁC TOÁN TỬ TRÊN CÁC KIỂU DỮ LIỆU

Toán tử	Mô tả
+	Phép cộng
-	Phép trừ
*	Phép nhân
/	Phép chia lấy phần nguyên
%	Phép chia lấy phần dư
-	Phép lấy số đối

Toán tử	Mô tả
+	Phép cộng
-	Phép trừ
*	Phép nhân
/	Phép chia
-	Phép lấy số đối

Biểu thức	Công dụng	Giá trị biểu thức
++x	Tăng x lên 1	Giá trị của x sau khi tăng
x++	Tăng x lên 1	Giá trị của x trước khi tăng
--x	Giảm x đi 1	Giá trị của x sau khi giảm
x--	Giảm x đi 1	Giá trị của x trước khi giảm

CÁC TOÁN TỬ TRÊN CÁC KIỂU DỮ LIỆU

Toán tử	Cách viết	Tương đương
+=	v += e;	v = v + e
-=	v -= e;	v = v - e
*=	v *= e;	v = v * e
/=	v /= e;	v = v / e
%=	v %= e;	v = v % e

Toán tử	Cách viết C++	Ý nghĩa
==	L == R	$L = R$
!=	L != R	$L \neq R$
<	L < R	$L < R$
>	L > R	$L > R$
<=	L <= R	$L \leq R$
>=	L >= R	$L \geq R$

Toán tử	Ký hiệu	Ký hiệu khác
Phủ định	!	not
"Và" logic	&&	and
"Hoặc" logic	||	or

- Các phép toán trên kiểu số nguyên
- Kiểu phép toán số thực
- Các phép toán logic

PHÂN TÍCH VÍ DỤ

```
1  | #include <iostream>
2  | using namespace std;
3  |
4  | int main()
5  | {
6  |     cout << boolalpha; //Yêu cầu in giá trị logic kiểu true/false thay vì 1/0
7  |     int i = 3;
8  |     bool b = (i < 1) && (++i == 4); //Bỏ qua phép so sánh ++i == 4
9  |     cout << b << '\n'; //in ra false
10 |     cout << i; //in ra 3, vì lệnh ++i ở trên không được thực hiện
11 | }
```

TOÁN TỬ ĐIỀU KIỆN (?)

- Toán tử điều kiện (?) đặt trong biểu thức có cú pháp sau:

  «Điều kiện» ? «A» : «B»;

- Trong đó «Điều kiện» là một biểu thức logic, «A» và «B» là hai biểu thức.
- Nếu điều kiện đúng, giá trị của cả biểu thức điều kiện được đặt bằng «A», ngược lại nếu điều kiện sai, giá trị của cả biểu thức điều kiện được đặt bằng «B».
- Ví dụ với x, y, z là ba biến kiểu int, muốn đặt z bằng giá trị lớn hơn trong hai giá trị x, y ($z = \max(x, y)$), ta có thể viết: z = x > y ? x : y;

THỨ TỰ ƯU TIÊN CỦA CÁC TOÁN TỬ

Thứ tự ưu tiên	Toán tử	Mô tả	Thứ tự
1	::	Phạm vi	→
2	++ --	Tăng/giảm: x++, x--	→
	()	Lời gọi hàm: f()	
	[]	Truy cập theo chỉ số: a[]	
	. ->	Truy cập thành viên	
3	++ --	Tăng/giảm: ++x, --x	←
	!	Phủ định logic: !x	
	~	Phép đảo bit: ~x	
	()	Toán tử ép kiểu: (T)x	
	- +	Toán tử số học một ngôi	
	& *	Lấy địa chỉ, hủy tham chiếu	
	new delete	Cấp phát, giải phóng bộ nhớ	
	sizeof()	Kích thước kiểu/biến	
4	.* ->*	Con trỏ tới thành viên	→
5	* / %	Phép toán nhân, chia, lấy dư	→
6	+ -	Phép toán cộng, trừ	→
7	<< >>	Dịch trái/phải dãy bit	→
8	< <= > >=	Quan hệ thứ tự	→
9	== !=	Quan hệ bằng/khác nhau	→
10	&	Toán tử BitAnd (hội bit)	→
11	^	Toán tử BitXor (hiệu đối xứng bit)	→
12	|	Toán tử BitOr (tuyển bit)	→
13	&&	Phép "và" logic	→
14	||	Phép "hoặc" logic	→
15	= *= /= %= += -= >>= <<= &= ^= |=	Phép gán/gán nhanh	←
	?	Toán tử điều kiện: a ? b : c	
16	,	Toán tử dấu phẩy	→

Biểu thức trong C++ có thể chứa rất nhiều toán tử và toán hạng, chúng ta cần biết được độ ưu tiên của các toán tử để hiểu cách thức trình dịch diễn giải biểu thức, qua đó đặt các cặp dấu ngoặc đơn hợp lý để biểu thức được tính đúng.

Ghi nhớ thứ tự ưu tiên của các toán tử liệt kê trong bảng bên.

ÉP KIỂU

- Chuyển kiểu ngầm định: một giá trị biểu thức được tự chuyển đổi sang kiểu khác nhằm thích nghi với việc tính toán hay gán giá trị cho biến. Việc này được trình biên dịch chuyển đổi ngầm định, tùy thuộc vào kiểu dữ liệu có trong biểu thức.
- Ép kiểu: C++ cho phép lập trình viên ép kiểu một số toán hạng để can thiệp vào quá trình ngầm chuyển đổi kiểu khi sinh mã tính toán biểu thức. Việc ép kiểu cho toán hạng a tương tự như việc lấy giá trị của a gán cho biến a' thuộc một kiểu khác rồi tính toán biểu thức theo toán hạng a' thay vì a.
- Có hai cú pháp đơn giản để ép kiểu:

  («Tên kiểu»)«Biểu thức» hoặc «Tên kiểu»(«Biểu thức»)

PHÂN TÍCH VÍ DỤ

```
1  #include<iostream>
2
3  using namespace std;
4
5  int main()
6  {
7      int a, b;
8      float kq;
9      cout << "Nhap a: "; cin >> a;
10     cout << "Nhap b: "; cin >> b;
11     if (!b) {//if(b==0)
12         cout << "Mau phai khac 0 !";
13         return 0;
14     }
15     //kq = (float)a / (float)b;
16     kq = float(a) / float(b);  // ep kieu
17     cout << "Ket qua la: " << kq << endl;
18     return 0;
19 }
```

```
Nhap a: 5
Nhap b: 14
Ket qua la: 0.357143
```

```
Nhap a: 10
Nhap b: 3
Ket qua la: 3.33333
```

Nếu không ép kiểu ở câu lệnh dòng 16, thì kết quả sẽ như thế nào?

CÂU HỎI KIỂM TRA

Các phép gán sau đây, phép nào hợp lệ? Vì sao?

```
x = 2 + 3*4;
2 + 3*4 = x;
x + 2 = y + 5;
x = y;
```

```
x = b*b – 4*a*c;
e = m*c2;
s = 3,14 * r * r;
c = 4 * 3.14 * r;
```

Trong các lệnh khai báo sau đây, hãy chỉ ra những lệnh KHÔNG hợp lệ và giải thích.

```
char x, y, z;
int x = 1; y = 2;
double x = y = 3.8;
unsigned int x = 1, y = 2;
unsignedshort x = 10;
```

```
const int x = 2;
const int x = y;
char ch = T;
char ch = “T”;
char ch = ‘T’;
```

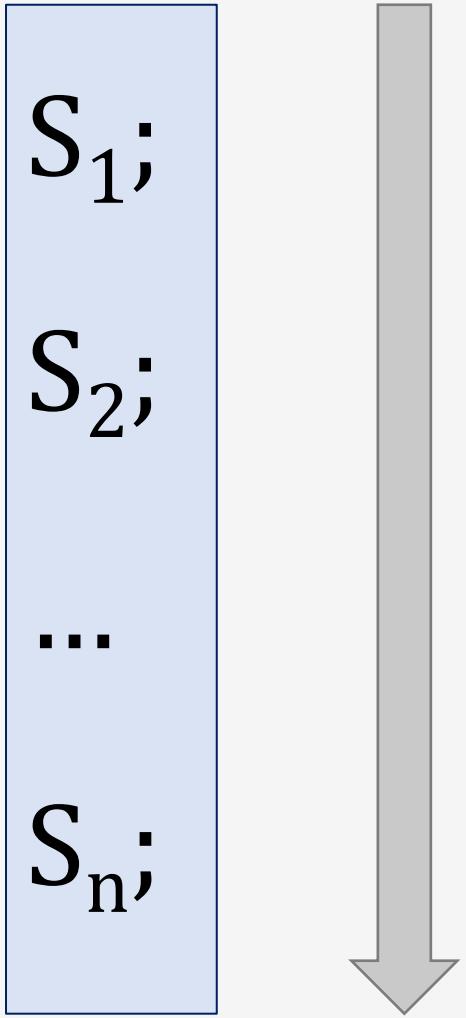
```
double t = 11;
int x = 4.7;
int x = ‘A’;
const int x;
char t = 65;
```

CÁC CẤU TRÚC LỆNH

- Cấu trúc tuần tự
- Lệnh ghép
- Cấu trúc rẽ nhánh
- Cấu trúc lặp
- Lệnh nhảy

CẤU TRÚC TUẦN TỰ

- S_1, S_2, …, S_n: là các lệnh
- Khi viết các lệnh liên tiếp chương trình sẽ thực hiện lần lượt các lệnh theo đúng thứ tự từ trên xuống dưới



VÍ DỤ 1-1

Nhập vào 3 số thực a, b, c là độ dài của 3 cạnh của một tam giác, hãy tính diện tích của tam giác đó theo công thức Heron.

$$p = \frac{a+b+c}{2}$$

$$s = \sqrt{p(p-a)(p-b)(p-c)}$$

VÍ DỤ 1-1 (vd1-1.cpp)

```
1  // tinh dien tich tam giac theo cong thuc Heron
2  #include <iostream>
3  #include <cmath>
4  using namespace std;
5  int main()
6  {
7     double a, b, c, p, s;
8     cout << "Cho ba canh: ";
9     cin >> a >> b >> c;
10    p = (a + b + c) / 2;
11    s = sqrt(p * (p - a) * (p - b) * (p - c));
12    cout << "Dien tich = " << s;
13 }
```

LỆNH GHÉP

- Có những nơi trong chương trình không cho phép viết nhiều hơn 1 lệnh, khi đó để thực hiện nhiều lệnh, cần phải "gói" chúng thành một lệnh ghép
- Lệnh ghép bắt đầu bằng "{", kết thúc bằng "}", giữa khối là các lệnh thành phần ngăn cách nhau bởi dấu chấm phẩy
- Dấu chấm phẩy sau S_n là thừa.
- Sau dấu đóng ngoặc nhọn không cần dấu ";" do C++ tự biết đây là nơi kết thúc lệnh

```
{
  S1;
  S2;
  ...
  Sn;
}
```

LỆNH if()

```
if (C) S;
```

- C: Điều kiện (Biểu thức logic), C có thể là biểu thức rỗng, có thể là giá trị hằng (true/false)
- S: Một lệnh hoặc nhiều lệnh, nếu muốn thực hiện nhiều lệnh, cần gói chúng vào thành lệnh ghép {}
- Khi gặp cấu trúc if, sẽ kiểm tra điều kiện C:
  - C = true, sẽ thực hiện lệnh S
  - C = false, không thực hiện lệnh

LỆNH RẼ NHÁNH if … elsse ….

```
if (C) S1;
else S2;
```

- C: Điều kiện (Biểu thức logic)
- S_1, S_2: Một lệnh hoặc nhiều lệnh, nếu muốn thực hiện nhiều lệnh, cần gói chúng vào thành lệnh ghép {}
- Khi gặp cấu trúc if…else…, sẽ kiểm tra điều kiện C:
  - C = true, máy thực hiện lệnh S_1
  - C = false, máy thực hiện lệnh S_2

VÍ DỤ 1-2

Nhập vào 3 số thực $a, b, c\ (a \neq 0)$

Hãy giải phương trình bậc 2: $ax^2 + bx + c = 0$

Thuật toán giải phương trình bậc hai theo delta.

Nhập a, b và c.

Tính $\Delta = b^2 - 4ac$

Nếu $\Delta < 0$, thông báo phương trình vô nghiệm

Nếu không...

Nếu $\Delta = 0$, đưa ra nghiệm kép $x = -\frac{b}{2a}$

Nếu không, thông báo 2 nghiệm $x = \frac{-b \pm \sqrt{\Delta}}{2a}$

VÍ DỤ 1-2 (vd1-2.cpp)

```cpp
// giai phuong trinh bac 2
#include <iostream>
#include <cmath>
using namespace std;
int main() {
  double a, b, c, delta;
  cout << "Cho a, b, c: ";
  cin >> a >> b >> c;
  delta = b * b - 4 * a * c;
  if (delta < 0) cout << "Vo nghiem!";
  else
    if (delta == 0) cout << "Nghiem kep x = " << -b / (2 * a);
    else {
      cout << "x1 = " << (-b - sqrt(delta)) / (2 * a) << endl;
      cout << "x2 = " << (-b + sqrt(delta)) / (2 * a) << endl;
    }
   return 0;
}
```

LỆNH LỰA CHỌN switch()

```
switch (e)
{
 case k1:
   GS1; break;
 case k2:
   GS2; break;
 …
 case kn:
   GSn; break;
 default:
   GDefault;
}
```

- e: Biểu thức kiểu đếm được: Số nguyên, ký tự, logic, liệt kê, đoạn con
- ki: hằng cùng kiểu với e
- GSi: Một nhóm lệnh
- Khi gặp lệnh switch, máy tính giá trị biểu thức e
  - Nếu giá trị e bằng ki, máy sẽ thực hiện nhóm lệnh GSi.
  - Nếu giá trị e không thuộc nhóm hằng nào, máy thực hiện nhóm lệnh GDefault, nếu không có phần GDefault, máy bỏ qua.
- break: Từ khóa của C++, sau mỗi nhóm lệnh cần có lệnh break để kết thúc.

PHÂN TÍCH VÍ DỤ

```
1  #include <iostream>
2  using namespace std;
3  int main() {
4        int n;
5        cout << "Nhap vao mot so [0, ... , 9] : ";
6        cin >> n;
7
8        switch (n) {
9        case 0: cout << n << " : Khong" << endl; break;
10       case 1: cout << n << " : Mot" << endl; break;
11       case 2: cout << n << " : Hai" << endl; break;
12       case 3: cout << n << " : Ba" << endl; break;
13       case 4: cout << n << " : Bon" << endl;  break;
14       case 5: cout << n << " : Nam" << endl; break;
15       case 6: cout << n << " : Sau" << endl; break;
16       case 7: cout << n << " : Bay" << endl; break;
17       case 8: cout << n << " : Tam" << endl; break;
18       case 9: cout << n << " : Chin" << endl;break;
19       default: cout << n << " phai nam trong [0, 9]" << endl;
20       }
21 return 0;
22 }
```

VÍ DỤ 1-3

Nhập vào 3 số nguyên, là ngày (d) tháng (m) và năm (y). Hãy cho biết ngày tháng năm đó là ngày thứ mấy. Biết rằng, công thức để tính thứ dựa trên ngày tháng năm như sau:

$$DOW = \left(d + \left\lfloor \frac{26(m+1)}{10} \right\rfloor + k + \left\lfloor \frac{k}{4} \right\rfloor + \left\lfloor \frac{c}{4} \right\rfloor + 5c\right) \% 7$$

0: Saturday, 1: Sunday, 2: Monday, …

Chú ý: các tham số trong công thức được giải thích như trong bảng

Tháng 1 và 2 coi như tháng 13 và 14 năm trước

18/2/2010 = 18/14/2009

Mỗi thế kỷ gồm 100 năm, các năm trong thế kỷ đánh số từ 0 tới 99: Thế kỷ 0: [0…99], Thế kỷ 1: [100…199], …, Thế kỷ 20: [2000, 2099]

c: Số hiệu thế kỷ

k: Số hiệu năm trong thế kỷ

VÍ DỤ 1-3 (vd1-3.cpp)

```cpp
#include <iostream>
using namespace std;
int main() {
  int d, m, y, c, k, dow;
  cout << "Cho biet (d, m, y): ";
  cin >> d >> m >> y;
  if (m < 3) {
    m += 12; y--;
  }
  c = y / 100;
  k = y % 100;
  dow = d + 26 * (m + 1) / 10 + k + k / 4 + c / 4 + 5 * c;
  switch(dow % 7) {
    case 0: cout << "Saturday"; break;
    case 1: cout << "Sunday"; break;
    case 2: cout << "Monday"; break;
    case 3: cout << "Tuesday"; break;
    case 4: cout << "Wednesday"; break;
    case 5: cout << "Thursday"; break;
    case 6: cout << "Friday"; break;
  }
return 0;
}
```

Nhập vào số nguyên dương n, hãy tính và đưa ra màn hình các tổng sau đây:

$$s1 = \frac{1}{1*2} + \frac{1}{2*3} + \cdots + \frac{1}{n*(n+1)}$$

$$s2 = 1 + \frac{1}{1+2} + \frac{1}{1+2+3} + \cdots + \frac{1}{1+2+3+\cdots+n}$$

LỆNH LẶP do ... while

```
do
  S;
while (C);
```

- Đây là vòng lặp không xác định với điều kiện sau
- C: Là điều kiện lặp (biểu thức logic)
- S: Một lệnh hoặc nhiều lệnh, nếu muốn thực hiện nhiều lệnh, phải "gói" các lệnh đó thành một lệnh ghép {}
- Máy sẽ thực hiện lệnh S và lặp lại lệnh này chừng nào điều kiện C đúng
- Lệnh S sẽ được thực hiện ít nhất 1 lần (Điều kiện lặp C được kiểm tra sau mỗi lượt lặp)
- Số lần lặp phụ thuộc vào giá trị của biểu thức logic C
- Chú ý: Trong S cần có ít nhất một lệnh làm thay đổi giá trị của C, nếu không sẽ lặp vô hạn.

VÍ DỤ 1-4

Cho một số nguyên N, tính tổng các chữ số trong biểu diễn thập phân của N?

Ý tưởng:

- Tách dần từng chữ số của N, từ chữ số hang đơn vị trở đi.
- Để tách được chữ số hàng đơn vị sử dụng phép toán chia lấy dư cho 10 (N%10)
- Để giảm N đi 1 chữ số, sử dụng phép chia lấy nguyên cho 10 (N/10).
- Quá trình trên lặp lại cho đến khi N = 0.
- Sử dụng 1 biến để cộng tổng các chữ số lại sẽ là kết quả.

VÍ DỤ 1-4 (vd1-4.cpp)

```cpp
// tinh tong cac chu so cua N
#include <iostream>
using namespace std;
int main() {
    long N, Sum;
    int d;
    cout << "N = ";
    cin >> N;
    Sum = 0;
    do
    {
        d = N % 10;   // lay chu so hang don vi cua N
        Sum = Sum + d;
        N = N / 10;   // giam N di 1 chu so
    } while (N != 0);
    cout << "Tong cac chu so la: " << Sum << endl;
    return 0;
}
```

LỆNH LẶP while

```
while (C)
 {
 S;
}
```

- Đây là vòng lặp không xác định với điều kiện trước
- C: Là điều kiện lặp (biểu thức logic)
- S: Một lệnh hoặc nhiều lệnh; trường hợp nếu S chỉ có 1 lệnh thì không cần có cặp {}
- Máy sẽ thực hiện lặp đi lặp lại lệnh S chừng nào điều kiện C đúng
- Nếu C sai ngay từ đầu, lệnh S sẽ không được thực hiện (điều kiện lặp C được kiểm tra trước mỗi lượt lặp)
- Số lần lặp phụ thuộc vào giá trị của biểu thức logic C
- Chú ý: Trong S cần có ít nhất một lệnh làm thay đổi giá trị của C, nếu không sẽ lặp vô hạn.

VÍ DỤ 1-5

Nhập vào hai số tự nhiên a, b. Tìm ước số chung lớn nhất của (a, b).

Ý tưởng:

Sử dụng thuật toán Euclide như sau:

- Lặp chừng nào a ≠ 0 và b ≠ 0;
- Nếu a > b thì thay thế a = a % b
- Ngược lại thay thế b = b % a.
- Kết thúc lặp khi một trong hai số a = 0 hoặc b = 0; nếu a = 0 thi b là USCLN và ngược lại.

VÍ DỤ 1-5 (vd1-5.cpp)

```cpp
// USCLN cua hai so nguyen
#include <iostream>
using namespace std;
int main()
{
  int a, b;
  cout << "a, b = "; cin >> a >> b;
  cout << "USCLN(" << a << ", " << b << ") = ";
  while ((a != 0) && (b != 0))
  {
    if (a > b) a = a % b;
    else b = b % a;
  }
  if (a == 0) cout << b << endl;
  else cout << a << endl;
  return 0;
}
```

LỆNH LẶP for

```
for (I; C; L) {
  S;
}
```

Tương đương với cấu trúc:

```
I;
while (C)
{
  S;
  L;
}
```

- Đây là vòng lặp với số lần lặp xác định
- I: Một biểu thức khởi tạo, thường là khởi tạo giá trị cho các biến điều khiển số lần lặp
- C: Điều kiện lặp (biểu thức logic)
- S: Một lệnh hoặc nhiều lệnh; nếu một lệnh thì không cần cặp {}
- L: Biểu thức sẽ được chạy sau mỗi bước thực thi lệnh S
- Ý nghĩa:
  - Máy chạy biểu thức khởi tạo I
  - Tại mỗi bước nếu điều kiện C là true, máy thực hiện lệnh S và sau đó là biểu thức L
  - Nếu điều kiện C là false, máy thoát vòng lặp for

PHÂN TÍCH CÁC VÍ DỤ

In lên màn hình các số có 2 chữ số.

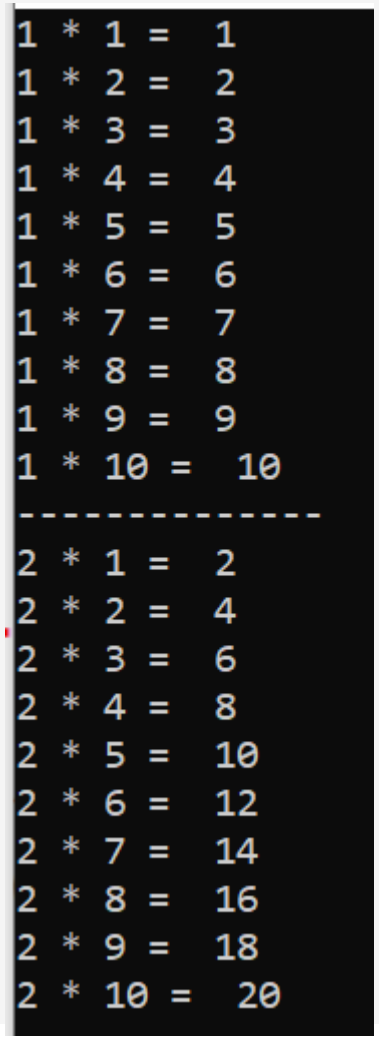
```
#include <iostream>
using namespace std;
int main()
{
 int i;
 for (i = 10; i < 100; i++)
   cout << i << " ";
 return 0;
}
```

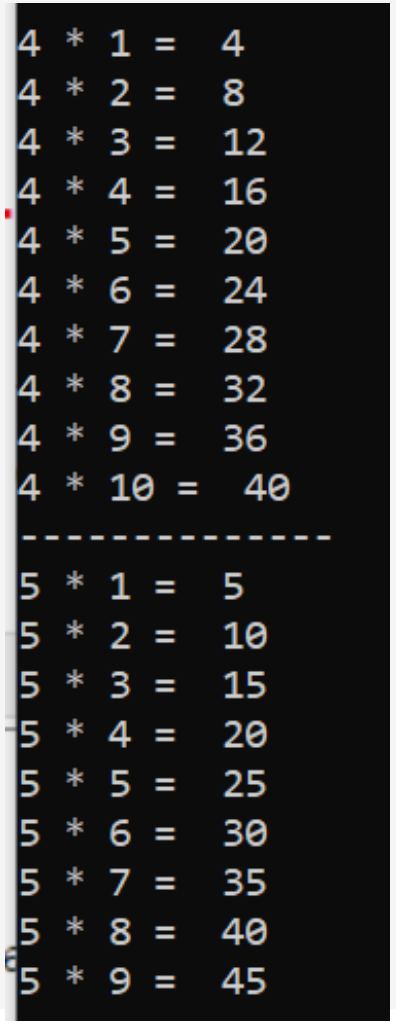
Nhập số nguyên dương n ≤ 20. Tính và đưa la màn hình giá trị n giai thừa.

```
#include <iostream>
using namespace std;
int main()
{
 int i, n;
 long f = 1;
 cout << "n = "; cin >> n;
 cout << n << "! = ";
 for (i = 1; i <= n; i++)  f = f * i;
 cout << f;
 return 0;
}
```

VÍ DỤ 1-6

.Viết chương trình in ra các bảng cửu chương từ 1 đến 9. Chẳng hạn như:





VÍ DỤ 1-6 (vd1-6.cpp)

```cpp
// in các bang cuu chuong
#include <iostream>
using namespace std;

int main()
{
    for (int n = 1; n <= 10; n++)
    {
        for (int i = 1; i <= 10; i++)
        {
            cout << n << " * " << i << " =  " << n * i << endl;
        }
        cout << "--------------" << endl; //Ket thuc mot bang
    }
    return 0;
}
```

LỆNH NHẢY goto

- Chọn một tên nhãn (chẳng hạn lab)
- Dùng nhãn lab và dấu ":" đánh dấu vị trí trước một lệnh trong chương trình.
- Lệnh
  - goto lab;
  - Khi gặp lệnh goto, máy sẽ nhảy tới vị trí đánh dấu trong bởi nhãn lab và thực hiện các lệnh từ vị trí đó.

```
int n, d, sum = 0;
cin >> n;
lab:
d = n % 10;
n /= 10;
sum += d;
if (n > 0) goto lab;
cout << sum;
```

Tính tổng các chữ số của số n; chú ý sử dụng lệnh goto thay cho lệnh lặp

LỆNH break VÀ continue

- Lệnh break;
  - Khi break được thực hiện trong một vòng lặp, vòng lặp đó sẽ ngưng ngay
  - Bản chất: goto tới vị trí sau vòng lặp
- Lệnh continue;
  - Khi continue được thực hiện trong một vòng lặp, toàn bộ lệnh trong vòng lặp phía sau continue sẽ bị bỏ qua, nhảy đến vị trí kiểm tra điều kiện lặp
  - Bản chất: goto tới vị trí kiểm tra điều kiện lặp

BÀI TẬP TẠI LỚP

Bài 1-1. Viết chương trình hiển thị lại bảng cửu chương theo dạng sau (không cần vẽ đường kẻ).

x	1	2	3	4	5	6	7	8	9	10
1	1	2	3	4	5	6	7	8	9	10
2	2	4	6	8	10	12	14	16	18	20
3	3	6	9	12	15	18	21	24	27	30
4	4	8	12	16	20	24	28	32	36	40
5	5	10	15	20	25	30	35	40	45	50
6	6	12	18	24	30	36	42	48	54	60
7	7	14	21	28	35	42	49	56	63	70
8	8	16	24	32	40	48	56	64	72	80
9	9	18	27	36	45	54	63	72	81	90
10	10	20	30	40	50	60	70	80	90	100

BÀI TẬP TẠI LỚP

Bài 1-2: Nhập vào số nguyên dương n, hãy tính và đưa ra màn hình các tổng sau đây (với kết quả có 3 chữ số sau dấu phẩy).

- $s1 = \frac{1}{1*2} + \frac{1}{2*3} + \cdots + \frac{1}{n*(n+1)}$
- $s2 = 1 + \frac{1}{1+2} + \frac{1}{1+2+3} + \cdots + \frac{1}{1+2+3+\cdots+n}$

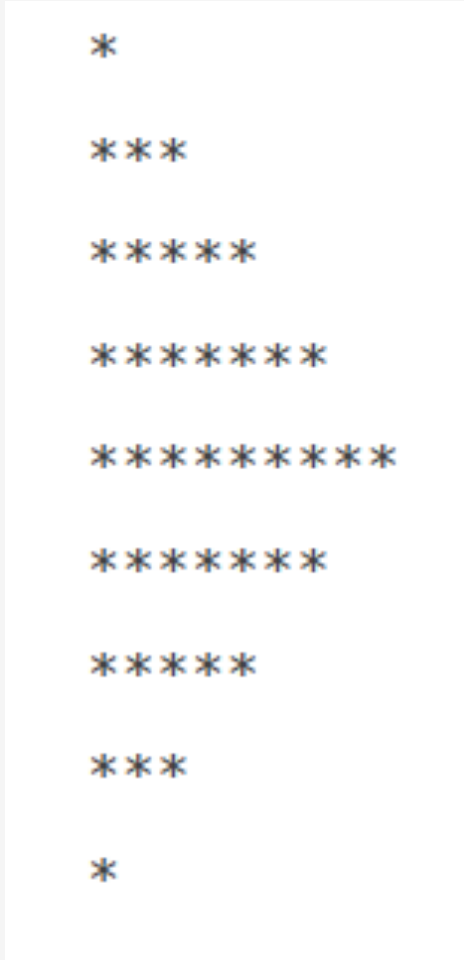
BÀI TẬP TẠI LỚP

.**Bài 1-3:** Viết chương trình cho phép người dùng nhập vào hai số m và y lần lượt là tháng và năm, in ra lịch của tháng đó. Ví dụ lịch tháng 9 năm 1945:

```
CN T2 T3 T4 T5 T6 T7
                   1
 2  3  4  5  6  7  8
 9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30
```

BÀI TẬP TẠI LỚP

Bài 1-4. Viết chương trình cho phép nhập vào từ bàn phím một số nguyên n và hiển thị ra màn hình hình dạng như hình bên với kích thước tùy theo giá trị nhập vào. Ví dụ ở hình bên với n = 5.



BÀI TẬP VỀ NHÀ

Bài 1-5: Viết chương trình cho nhập vào ba số thực a, b, c là độ dài 3 cạnh của một tam giác, tính toán và in ra: kết quả với 4 chữ số sau dấu chấm thập phân.

- Diện tích tam giác đó theo công thức
- Bán kính đường tròn ngoại tiếp
- Nội tiếp của tam giác

Bài 1-6: Hai số nguyên dương a, b được gọi là "bạn bè" nếu tổng các ước dương thực sự của a bằng b và tổng các ước dương thực sự của b bằng a. Hãy liệt kê tất cả các cặp số bạn bè trong phạm vi từ [1; 10^6].