NHẬP MÔN LẬP TRÌNH

CHƯƠNG 3.1: CÁC KIỂU DỮ LIỆU CƠ BẢN

FUNDAMENTAL DATA TYPES

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

Nội dung

1. Bộ từ vựng (**keywords**) trong C++
2. Các kiểu dữ liệu cơ sở (**Fundamental data types**)
3. Biến (**Variable**)
4. Hằng (**Constant**)

Bài tập minh họa

1. Bộ từ vựng trong C++

1. Ký tự (**Character**)
2. Từ khóa (**Keyword**)
3. Dấu chấm phẩy
4. Tên/ Định danh (**Name/Indetifier**)
5. Hằng (**Constant**)
6. Câu chú thích (**Comment**)

1.1. Ký tự

- **Bộ chữ cái 26 ký tự Latin**
  A, B, C, …, Z
  a, b, c, …, z

- **Bộ chữ số thập phân**
  0, 1, 2, …, 9

- **Các ký hiệu toán học**

+ – * / = < > ( )

- **Các ký tự đặc biệt**

. , : ; [ ] % \ # $ ‘’_ và khoảng trắng

Lưu ý:
Khi viết chương trình C++ chỉ sử dụng các ký tự này.

1.2. Từ khóa

- Không thể sử dụng từ khóa để đặt tên cho biến, hàm, tên chương trình con.
- Viết bằng chữ thường.

Một số từ khóa thông dụng:

const, enum, signed, struct, typedef, unsigned

char, double, float, int, long, short, void

case, default, else, if, switch

do, for, while

break, continue, goto, return

1.3. Dấu chấm phẩy

- Dùng để phân cách các câu lệnh.
- Các câu lệnh có thể viết trên 1 dòng, tuy nhiên luôn được phân cách bằng dấu chấm phẩy ;

```
cout << "Xin chao"; return 0;
```

1.4. Tên/ Định danh (Identifier)

- Một dãy ký tự dùng chỉ tên hằng, biến, kiểu dữ liệu, hàm
- Được tạo thành từ các chữ cái, chữ số, dấu gạch nối _
- Qui ước đặt tên:
  - Không trùng với các từ khóa
  - Ký tự đầu tiên là chữ cái hoặc _
  - Tối đa 255 ký tự
  - Không được sử dụng khoảng trắng ở giữa các ký tự
  - Phân biệt chữ hoa chữ thường (case sensitive).
- Một số ví dụ sai khi đặt tên:

1abc	default	hello world
@mail	case	
X-1	x&y	
f(x)	ho-ten	

? Các bộ tên sau đây có giống nhau không?

- A, a
- BaiTap, baitap, BAITAP, bAItaP, …

1.5. Hằng

- **Hằng số:**

  1, 200

- **Hằng ký tự:**

  ‘A’, ‘a’

- **Hằng chuỗi:**

  “Xin Chao”

Chú ý: ‘A’ khác “A”

- …

1.6. Câu chú thích

- Dùng để mô tả hoặc ghi chú trong source code
- Giúp dễ dàng đọc code sau này
- Có 2 cách để viết chú thích trong C++

Cách 1: Viết chú thích trong /*chú thích*/

```
/* Day la cau chu thich 1
   Day la cau chu thich 2*/
```

Cách 2: Viết chú thích sau // chú thích

```
// Day la cau chu thich 1
// Day la cau chu thich 2
```

- Nên sử dụng nhất quán 1 cách. Cách 2 được sử dụng phổ biến hơn
- Khi biên dịch source code thì các phần comments sẽ không được biên dịch

2. Các kiểu dữ liệu cơ sở

1. Kiểu ký tự (**Character types**)
2. Kiểu số nguyên (**Numerical Integer types**)
3. Kiểu số thực (**Floating-point types**)
4. Kiểu luận lý/logic (**Boolean type**)
5. Kiểu void (**void type**)
6. Nội dung khác: Giới thiệu về kiểu phức hợp (**compound data types**): kiểu string, Khai báo Typedef (**typedef Declarations**) , Enum
7. Một số hàm hữu ích

Giới thiệu

- C++ có các kiểu cơ sở như sau:

STT	Tên kiểu	Tên kiểu	Ví dụ
1	Kiểu ký tự Character types	char, …	'A', '? ‘, '1', …
2	Kiểu số nguyên Numerical Integer types	int long …	10, 232, …
3	Kiểu số thực Floating-point types	float double long double	3.1415, -29.12, …
4	Kiểu luận lý Boolean type	bool	true, false
5	Kiểu void Void type	void	*Không lưu giá trị*
6	Null pointer (C++11)	decltype(nullptr)	nullptr

http://www.cplusplus.com/doc/tutorial/variables/

2.1 Kiểu ký tự

Kiểu dữ liệu	Kích thước	Giá trị
char	1 byte	[-128; 127] hoặc [0; 255]
unsigned char	1 byte	[0; 255]

- Biểu diễn thông qua bảng mã **ASCII**. Bảng mã ASCII chuẩn có 128 ký tự, bảng mã ASCII mở rộng có 256 ký tự bao gồm 128 ký tự trong bảng mã ASCII chuẩn và ký tự đồ họa.

https://en.wikipedia.org/wiki/ASCII

2.1 Kiểu ký tự

- Các ký tự có mã nhỏ hơn thì nhỏ hơn

```
#include <iostream>
int main() {
    char a = 'a';
    char b = 'b';
    std::cout << (a<b);
    return 0;
}
```

```
1
```

- Phân loại 256 ký tự thành 3 nhóm:
  - Ký tự điều khiển (ASCII chuẩn): 0 → 31 và 127 (non-printable)
  - Ký tự văn bản (ASCII chuẩn): 32 → 126
  - Ký tự đồ họa (extended ASCII): 128 → 255

ASCII chuẩn

*	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	TAB	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2	space	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL

OEM Extended ASCII

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8	Ç	ü	é	â	ä	à	å	ç	ê	ë	è	ï	î	ì	Ä	Å
9	É	æ	Æ	ô	ö	ò	û	ù	ÿ	Ö	Ü	¢	£	¥	₧	ƒ
A	á	í	ó	ú	ñ	Ñ	ª	º	¿	⌐	¬	½	¼	¡	«	»
B	▒	▓	▒	|	┤	╡	╢	╖	╕	╣	╗	╝	╜	┘	┐	
C	└	┴	┬	├	─	┼	╞	╟	╚	╔	╩	╦	╠	═	╬	╧
D	╨	╤	╥	╙	╘	╒	╓	╫	╪	┘	┌	▀	▄	▌	▐	■
E	α	ß	Γ	π	Σ	σ	µ	τ	Φ	Θ	Ω	δ	∞	ø	∈	∩
F	≡	±	≥	≤	⌠	⌡	÷	≈	°	∙	·	√	ⁿ	²	■	

ANSI Extended ASCII

ANSI Extended ASCII (Windows)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8	□	□	‚	ƒ	„	…	†	‡	ˆ	‰	Š	‹	Œ	□	□	□
9	□	‘	’	“	”	•	–	—	˜	™	š	›	œ	□	□	Ÿ
A		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	-	®	¯
B	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
C	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
E	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
F	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

Chương trình xuất mã ASCII cho 1 ký tự nhập từ bàn phím

```
#include <iostream>
int main() {
    char ascii;
    while(1) {
        std::cout << "Nhap ky tu : ";
        std::cin >> ascii;
        std::cout << "Ma ASCII la: « << (int)ascii
                  << std::endl;
    }
    return 0;
}
```

2.2 Kiểu số nguyên

Kiểu dữ liệu (Type)	Kích thước (Size)	Phạm vi (Range of Value)
signed **char**	1 byte	[-128; 127]
signed **short** int	2 bytes	[-32,768; 32,767]
signed **int (** hoặc **signed**)	4 bytes	[-2,147,483,648; 2,147,483,647]
signed **long**	4 bytes	[-2,147,483,648; 2,147,483,647]
signed **long long** int	8 bytes	[-9,223,372,036,854,775,808; 9,223,372,036,854,775,807]
unsigned char	1 byte	[0; 255]
unsigned short int	2 bytes	[0; 65,535]
unsigned int	4 bytes	[0; 4,294,967,295]
unsigned long int	4 bytes	[0; 4,294,967,295]
unsigned long long int	8 bytes	[0; 18,446,744,073,709,551,615]

- Tìm hiểu thêm cách viết khác nhau của cùng một kiểu dữ liệu. Ví dụ: 4 cách viết (signed **short** int, **sort** int, signed **sort, sort**) đều mang ý nghĩa như nhau.
- Các kiểu số nguyên (có dấu)
  n bit có dấu: -2^{n-1} … $+2^{n-1} - 1$
- Các kiểu số nguyên (không dấu)
  n bit không dấu: 0 … $2^n - 1$

int vs long ???

Kích cỡ của kiểu dữ liệu phụ thuộc vào compiler và cấu hình máy tính đang sử dụng. Ví dụ: Kích cỡ kiểu **long** như bên

OS	arch	size
Windows	IA-32	4 bytes
Windows	Intel 64	4 bytes
Windows	IA-64	4 bytes
Linux	IA-32	4 bytes
Linux	Intel 64	8 bytes
Linux	IA-64	8 bytes
Mac OS X	IA-32	4 bytes
Mac OS X	Intel 64	8 bytes

- Toán tử **sizeof** dùng để kiểm tra kích thước của kiểu dữ liệu.

```
//Chương trình kiểm tra kích thước kiểu dữ liệu:
#include <iostream>
int main() {
    std::cout << sizeof(char)  << " byte\n";
    std::cout << sizeof(short) << " bytes\n";
    std::cout << sizeof(int)   << " bytes\n";
    std::cout << sizeof(long)  << " bytes\n";
    std::cout << sizeof(long long)  << " bytes\n";
    return 0;
}
```

Các cách biểu diễn số thực:

1. **Dạng thập phân:**

   45.0 -256.45 +122.8 .34 15.

2. **Dạng khoa học:**

$$1.257E + 01 = 1.257 * 10^{1} = 12.57$$
$$1257.0E - 02 = 1257 * 10^{-2} = 12.57$$

Kiểu dữ liệu	Kích thước	Phạm vi
float	4 bytes	[1.17549E-38, 3.40282E+38](~6 chữ số)
double	8 bytes	[2.22507E-308,1.79769E+308](~15 chữ số)
long double	12 bytes	[3.3621E-4932, 1.1893E+4932](~18 chữ số)

For floating-point types, the size affects their precision, by having more or less bits for their significant and exponent.

double vs long double ???

https://en.wikipedia.org/wiki/Long_double

Ví dụ

```
#include <iomanip> // std::setprecision
#include <iostream> // std::cout
#include <cfloat> // LDBL_DIG

int main() {
   float PI_float; double PI_double; long double PI_ldouble;
   PI_float  = 3.14159265358979323846264338327950288419716939937510L;
   PI_double   = 3.14159265358979323846264338327950288419716939937510L;
   PI_ldouble = 3.14159265358979323846264338327950288419716939937510L;
   std::cout << "PI = 3.14159265358979323846264338327950288419716939937510"<<endl;
   std::cout << std::setprecision(LDBL_DIG+3) << "PI_float = " << PI_float << endl;
   std::cout << std::setprecision(LDBL_DIG+3) << "PI_double = " << PI_double << endl;
   std::cout << std::setprecision(LDBL_DIG+3) << "PI_ldouble = " << PI_ldouble << endl;
}
```

Kết quả: (Window x64, GNU GCC++ Compiler in IDE Code-Blocks)

PI = 3.**14159265358979323846**26433832795028841971693993751**0**

PI_float = 3.**14159**27410125732421875

PI_double = 3.**14159265358979311599796346854418516159057617187**5

PI_ldouble = 3.**14159265358979323851280895940618620443274267017**84

2.4 Kiểu luận lý/logic

Kiểu dữ liệu	Kích thước	Giá trị
bool	1 byte	**false**: giá trị 0 **true**: giá trị khác 0

```
bool isTrue1 = 1;
bool isFalse1 = 0;
bool isTrue2 = true;
bool isFalse2 = false;
```

2.5 Kiểu void

- Kiểu dữ liệu rỗng không chứa gì cả
- Có 2 cách sử dụng:

Cách 1: Giá trị trả về cho hàm. Khi không cần giá trị trả về

```
void swap(int &a, int &b) {
    int c = b;
    b = a;
    a = b;
}
```

Cách 2: Một con trỏ chung không trỏ về bất kì giá trị nào

```
int a = 10;
float b = 5;
void *c;
c = &a;
c = &b;
```

2.6 Giới thiệu kiểu string

- Một trong những điểm mạnh của C++ là nó cung cấp những kiểu dữ liệu kết hợp dựa trên những kiểu cơ bản.
- Biến có kiểu string có thể chứa một chuỗi các ký tự (gọi là chuỗi).
- Ví dụ:

```
#include <iostream>
#include <string>
int main () {
	string mystring;
	mystring = "This is the initial string content";
	std::cout << mystring << std::endl;
	mystring = "This is a different string content";
	std::cout << mystring << std::endl;
	return 0;
}
```

2.6 Typedef

typedef dùng để đặt tên mới cho một kiểu dữ liệu có sẵn

Cấu trúc:

```
typedef   kiểu_có_sẵn   tên_mới;
```

Ví dụ:

```
typedef int songuyen;
int a;
songuyen b;
```

2.6 Enum

Enum là kiểu dữ liệu giúp hỗ trợ định nghĩa những giá trị liệt kê.

Cấu trúc:

```
enum tên_danh_sách {danh_sách_các_tên};
```

Ví dụ:

```
#include <iostream>
enum gioi_tinh {
    nam = 1,
    nu = 2,
    khac = 3,
};

int main() {
    gioi_tinh sv_gt = nam;
    std::cout << sv_gt;
    return 0;
}
```

Tại sao nên sử dụng enum?

1. Nhất quán
2. Source code rõ ràng
3. Dễ

   nâng cấp,

   sửa chửa,

   bảo trì.

2.7 Một số hàm hữu ích xử lý kiểu dữ liệu

Thư viện	Ví dụ	Kết quả
<limits>	std::numeric_limits<int>::min()	-2147483648
	std::numeric_limits<int>::max()	2147483647
	std::numeric_limits<float>::max()	3.40282e+038
	std::numeric_limits<float>::digits10	6
	…	
<climits>	INT_MIN	-2147483648
	INT_MAX	2147483647
	…	
<cfloat>	FLT_MAX	3.40282e+038
	FLT_DIG	6
	…	

3. Biến

1. Giới thiệu chung về biến
2. Cú pháp khai báo biến
3. Địa chỉ của biến
4. Vị trí khai báo biến
5. Biến cục bộ (**local variable**)
6. Biến toàn cục (**global variable**)
7. Khởi tạo biến cục bộ và toàn cục
8. Các loại biến

Sự khác nhau giữa khái niệm biến trong toán học và biến trong lập trình?

3.1 Giới thiệu chung

- Biến là **một ô nhớ hoặc một vùng nhớ dùng để chứa dữ liệu trong quá trình thực hiện chương trình**
- Mỗi biến có một kiểu dữ liệu cụ thể, kích thước của biến phụ thuộc vào kiểu dữ liệu.
- Giá trị của biến có thể được thay đổi. Trước khi sử dụng phải được khai báo. (WHY?)
- Quy cách đặt tên biến:
  - Không trùng với các từ khóa, hoặc tên hàm.
  - Ký tự đầu tiên là chữ cái hoặc _
  - Không được sử dụng khoảng trắng ở giữa các ký tự
  - *Nên sử dụng tất cả chữ thường với dấu _ giữa các từ.*

```
int so_nguyen;
float so_thuc;
```

https://google.github.io/styleguide/cppguide.html

3.2 Khai báo biến

- Cú pháp khai báo biến:

  - Cách 1: `kiểu_dữ_liệu tên_biến_1,tên_biến_2;`
  - Cách 2:

```
kiểu_dữ_liệu tên_biến_1;
kiểu_dữ_liệu tên_biến_2;
```

```
int     i, j, k;
char    c, ch;
float   f, salary;
double  d;
```

Ví dụ:
Viết chương trình nhập vào 3 số a,b và c.
Cho biết a, b ,c có tạo thành 3 cạnh của tam giác không ?

Cần khai báo bao nhiêu biến?

```
int a;
int b;
int c;
```

```
int a,b,c;
```

3.2 Khai báo biến

- Trong C++ có 3 cách **khởi tạo giá trị** cho biến:

```
int NamSinh =1997; // Copy initialization
int NamSinh (1997); //direct initialization
int NamSinh {1997}; // Uniform initialization C++11
```

- Phép gán: Sử dụng toán tử gán

```
int NamSinh;
NamSinh=1997;
```

3.3 Địa chỉ của biến

RAM được tạo nên bởi nhiều ô nhớ.

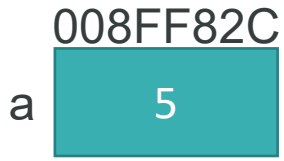
Mỗi ô nhớ có kích thước 1 byte.

Mỗi ô nhớ có địa chỉ duy nhất và được đánh số từ 0 trở đi.

Mỗi 1 biến khi được khai báo sẽ được cấp 1 vùng nhớ với địa chỉ duy nhất để lưu trữ biến đó.

Để truy cập vào địa chỉ của một biến ta sử dụng toán tử &.

```
#include <iostream>
int main() {
    int a = 5;
    std::cout << "Gia tri cua a: « << a << '\n';
    std::cout << "Dia chi cua a: « << &a<< '\n';
    return 0;
}
```




```
Gia tri cua a: 5
Dia chi cua a: 008FF82C
```

Quy trình xử lý biến của Trình biên dịch

- Dành riêng một vùng nhớ với địa chỉ duy nhất để lưu biến đó
- Liên kết địa chỉ vùng nhớ đó với tên biến
- Khi gọi tên biến, nó sẽ truy xuất tự động đến vùng nhớ đã liên kết với tên biến

3.4 Vị trí khai báo biến

- C bắt buộc khai báo tất cả các biến ở đầu một hàm.
- C++ có thể định nghĩa các biến ở **bất kỳ vị trí nào** trong hàm.

3.5 Biến cục bộ

- Biến được **định nghĩa** trong một hàm hoặc 1 block được gọi là biến cục bộ
- Biến cục bộ chỉ được sử dụng bên trong hàm hoặc block
- Các hàm bên ngoài khác sẽ không truy cập được biến cục bộ

```
#include <iostream>
int main() {
    int a = 2, b = 3;
    int c;
    c = a + b;
    std::cout << c;
    return 0;
}
```

Biến a, b, c là các biến cục bộ bên trong hàm main.

3.6 Biến toàn cục

- Biến toàn cục được định nghĩa bên ngoài các hàm, và thường được định nghĩa ở phần đầu của source code file.
- Biến toàn cục sẽ giữ giá trị của biến xuyên suốt chương trình.
- Tất cả các hàm đều có thể truy cập biến toàn cục

Ví dụ 1:

```
#include <iostream>
int g;
int main() {
    int a = 2, b = 3;
    g = a + b;
    std::cout << g;
    return 0;
}
```

g = ???

Ví dụ 2:

```
#include <iostream>
int g = 20;
int main() {
    int a = 2, b = 3;
    g = a + b;
    std::cout << g;
    return 0;
}
```

g = ???

3.7 Khởi tạo biến toàn cục và cục bộ

- Khi biến cục bộ được định nghĩa, giá trị của biến sẽ không được khởi tạo.

→ **Ta phải gán giá trị cho biến cục bộ để khởi tạo**

- Biến toàn cục sẽ được **tự động khởi tạo** giá trị.

Kiểu dữ liệu	Giá trị khởi tạo
int	0
char	'\0'
float	0
double	0
pointer	NULL

3.8 Các loại biến

- Các loại biến: biến thường, biến con trỏ (con trỏ chỉ là 1 biến và giá trị của nó là địa chỉ), biến tham chiếu (biến này cũng chỉ là 1 biến nhưng nó không được cấp phát ô nhớ riêng và dung chung ô nhớ với biến gốc) (phần biến con trỏ và biến tham chiếu sẽ nói chi tiết thêm trong các phần sau)

Ví dụ

Cho biết giá trị của biến khi chạy tuần tự từng dòng lệnh bên dưới đây:

```
int x;
int y=5;
int z=y;
x=y;
int*p=&x;
int &m=x;
```

4. Hằng (constant)

- Hằng đại diện cho một giá trị không đổi trong suốt quá trình thực thi của chương trình.
- Không thể gán lại giá trị cho hằng.

Loại hằng	Ví dụ
Hằng số nguyên Integer literal	-1090L, 212, 0213, 0x4b 18, 18u, 18l, 18ll, 18ul, 18ull // u or U: unsigned, l or L: long, ll or LL: long long (upper or lowercase letters is the same)
Hằng số thực - Floating-point literal	102.0, -223.1, 1234.56*e*−3, … 3.14159L, 6.02e23f // l or L: long double, f or F: float(upper or lowercase letters is the same)
Hằng luận lý	true, false
Hằng ký tự, hằng chuỗi Char literal, String literal	65, 0101, 0x41 hoặc 'A' , '\101' , '\x41‘ (hằng ký tự ‘A’) '\0‘, '\x0’ (dấu kết thúc câu) 27 (ký tự Escape) “C:\user\username\local”, …
Hằng con trỏ Poiter literal	…
Hằng kiểu dữ liệu được định nghĩa User-defined literal	…

4. Hằng

Cách định nghĩa hằng trong C++: Có 2 cách

Cách 1: Sử dụng tiền xử lý (Preprocessor definitions) **#define**

Câu lệnh:

```
#define tên_hằng giá_trị
```

Lưu ý: Không có ký tự ;

Ví dụ:

```
#include <iostream>
#define PI 3.14
int main() {
    int r = 2;
    std::cout << 2*r*PI;
    return 0;
}
```

```
6.28
```

Cách định nghĩa hằng trong C++: Có 2 cách

Cách 2: Sử dụng **const**

Câu lệnh:

```
const kiểu_giá_trị tên_hằng = giá_trị_hằng;
```

Ví dụ:

```
#include <iostream>
int main() {
    const float PI = 3.14;
    int r = 2;
    std::cout << r*PI;
    getch();
    return 0;
}
```

6.28

Bài tập về nhà:

Câu lệnh #define và const khác nhau thế nào?

Bài tập

- Chọn khai báo đúng:

1. Biến HoTen có giá trị là "Sau Rieng"
   a. string Ho-Ten = "Sau Rieng";
   b. string HoTen = "Sau Rieng";
   c. string HoTen = Sau Rieng;

2. Biến GioiTinh có giá trị là 'F'
   a. char GioiTinh('F');
   b. Char GioiTinh = 'F';
   c. int GioiTinh = 'F';

3. Biến lưu trữ Điểm toán là 10 và biến điểm lý là 9.
   a. int Diem_Toan(10), DiemLy{9};
   b. float Diem_Toan(10), DiemLy[9];

4. Điểm trung bình là trung bình cộng điểm toán cộng điểm lý
   a. float DiemTrungBinh = Diem_Toan + DiemLy / 2;
   b. float DiemTrungBinh = (Diem_Toan + DiemLy) / 2;

Bài tập

- Khai báo biến sau:

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)

Bài tập

- Sửa lại đoạn khai báo biến sau:

```
int 1HoTen;
float Gioi-Tinh;
bool Diem_Toan, DiemLy;
char DiemTrungBinh;
int _XepLoai;
string KetQua;
```

=> Viết chương trình khai báo các biến dữ liệu như bên.

Bài tập

Bài tập

Khai báo hằng ký hiệu MAX có giá trị là 1000 sử dụng tiền chỉ thị #define

Khai báo hằng ký hiệu MIN có giá trị là 0 sử dụng từ khóa const

```
#define MAX 1000
const int MIN = 0;
```

Bài tập minh họa

1. Cho biết năm sinh của một người và tính tuổi của người đó.
2. Cho 2 số a, b. Tính tổng, hiệu, tính và thương của hai số đó.
3. Cho biết 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. Cho biết điểm thi và hệ số 3 môn Toán, Lý, Hóa của một sinh viên. Tính điểm trung bình của sinh viên đó.
5. Cho biết bán kính của đường tròn. Tính chu vi và diện tích của hình tròn đó.

Cho biết năm sinh của một người và tính tuổi của người đó?

```
#include <iostream>
int main() {
    int namsinh = 1998;
    int tuoi = 0;
    tuoi = 2016-namsinh;
    return 0;
}
```

Bài tập 2

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

```
#include <iostream>
int main() {
    int a = 10;
    int b =  3;
    int tong = 0;
    int hieu = 0;
    int tich = 0;
    float thuong = 0;
    tong = a + b;
    hieu = a - b;
    tich = a * b;
    thuong = (float)a / b;
    return 0;
}
```

Bài tập 3

Cho biết 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 Soluong = 10;
    int Dongia = 500;
    int Tien = 0;
    float Vat = 0;
    Tien = Soluong * Dongia;
    Vat = Tien * 0.1;
    return 0;
}
```

Bài tập 4

Cho biết điểm thi và hệ số 3 môn Toán, Lý, Hóa của một sinh viên. Tính điểm trung bình của sinh viên đó.

```
#include <iostream>
int main() {
    float toan = 6.5;
    float hstoan = 2.0;
    float ly = 7.0;
    float hsly = 1.0;
    float hoa = 7.5;
    float hshoa = 1.0;
    float Dtb = 0;
    Dtb = (toan * hstoan + ly * hsly + hoa * hshoa) / (hstoan + hsly + hshoa);
    return 0;
}
```

Bài tập 5

Cho biết 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 = 6.5;
    float chuvi = 0;
    float dientich = 0;
    chuvi = 2 * PI * r;
    dientich = PI * r * r;
    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. Cho số thực x. Tính giá trị các biểu thức sau:

a)

$$y_1 = 4(x^2 + 10x\sqrt{x} + 3x + 1)$$

b)

$$y_2 = \frac{\sin(\pi x^2) + \sqrt{x^2 + 1}}{e^{2x} + \cos\left(\frac{\pi}{4}x\right)}$$

5. Viết chương trình cho 2 giờ (giờ, phút, giây) và thực hiện cộng, trừ 2 giờ này.

Chúc các em học tốt!