**Ngôn Ngữ Lập Trình C**

**KIẾN THỨC CƠ BẢN C**

**1.Thân chương trình chính**

#include <stdio.h>

#include <tên thư viện>

int main () {

         //code;

}

**2.Nhập xuất C**

|  |  |
| --- | --- |
|  | Ngôn ngữ lập trình c |
| Nhập từ bàn phím | Scanf (“đặc tả”, **&**tên\_biến); |
| Xuất ra màn hình | Printf (“đặc tả + nội dung”, ten\_bien); |

**3.Các đặc tả trong C**

|  |  |
| --- | --- |
| Kiểu dữ liệu | Đặc tả |
| int | %d |
| float | %f |
| double | %llf |
| Long long | %lld |
| char | %c |
| Mảng kí tự | %s |

Ví dụ Nhập a và b xuất ra màn hinh “gia\_tri a+gia\_tri b=gia tri\_a+b”

|  |
| --- |
| Ngôn ngữ C |
| #include <stdio.h>  Int main () {           Int a, b, c;           Scanf (“%d %d %d, &a, &b, &c);           Printf (“%d + %d = %d, a, b, c);  } |

**4.Các kiểu dữ liệu**

|  |  |
| --- | --- |
| Kiểu dữ liệu | Kích thước và miền giá trị biến |
| int | 4 byte (-10^9 đến **10^9)** |
| Long long | 8 byte (-10^18 đến 10^18) |
| float | 4 bytes (Lấy tối đa **6 chữ số** sau dấu phẩy) |
| Double | 8 bytes (Lấy tối đa **15 chữ số** sau dấu phẩy) |
| Char | 1 byte (0 đến 255) |

**Một số bảng mã Asscii thường gặp**

|  |  |  |
| --- | --- | --- |
| **Ký tự in thường** | **Ký tự in hoa** | **Kí tự số** |
| 97 đến 122 | 65 đến 90 | 48 đến 57 |

*In thường về in hoa -32*

*Từ in hoa về in thường +32*

**5.Khai báo biến**

Kieu\_du\_lieu tenbien;

*Ví dụ:*

int a, banKinh, dienTich;

float b;

**5.Các Quy Tắc Đặt Tên Biến**

- Không được bắt đầu bằng chữ số

- Không có dấu cách

- Không trùng với các keyword trong c/c++.

**6.Comment trong C/C++**

|  |  |
| --- | --- |
| //: Comment trên dòng | /\*           codes (nhiều dòng)  \*/ |

**7.Xuống dòng trong c và c++**

|  |
| --- |
| Ngôn ngữ c |
| Printf (“\n”); |

**8. Khởi tạo giá trị cho biến**

Kieu du lieu ten\_bien = gia tri;

**9. Lấy độ chính xác sau phần thập phân**

|  |
| --- |
| Ngôn ngữ c |
| Đặc tả: %.nf với n là số lượng chữ số sau dấu phẩy |

**10. Toán tử gán**

Ten bien = ten bien/giá trị

Kieu du lieu ten\_bien = gia tri/ten\_bien

**11. Toán tử số học “+”, “- “, “\*”, “/”, “%”**

|  |  |
| --- | --- |
| Toán tử | Ý nghĩa |
| “+” | Toán tử cộng |
| “- “ | Toán tử trừ |
| “\*” | Toán tử nhân |
| “/” | Toán tử chia lấy nguyên |
| “%” | Toán tử chia lấy dư |
| “++” | Toán tử tăng 1 đơn vị |
| “- -” | Toán tử giảm 1 đơn vị |

***Lưu ý***

Int b=a++; (tăng sau)

Int c=++a; (tăng trước)

b=a

c=a+1;

a+=1 <-> a=a+1 <-> a++;

**12.Ép kiểu trong c++/c**

Lưu ý khi thực hiện phép chia phải ép kiểu

Ví dụ

Int a, b;

Float s=(float)a/b; //hoặc s=1,0\*a/b;

Float s1=a/b;

Kết quả

A=5, b=2 -> s=2.5, s1=2;

*(Kiểu dữ liệu) tên biến cần ép*

*Ví dụ*

1ll\*tên biến.

1.0\*tên biến.

**13.Toán tử so sánh**

|  |  |
| --- | --- |
| Toán tử | Ý nghĩa |
| > | Lớn hơn |
| < | Bé hơn |
| >= | Lớn hơn hoặc bằng |
| <= | Bé hơn hoặc bằng |
| != | Khác |
| == | So sánh bằng |

**14.Toán tử nối**

|  |  |
| --- | --- |
| Toán tử | Ý nghĩa |
| && | And |
| || | Or |
| ! | Not |

**CÂU LỆNH RẼ NHÁNH**

**Câu lệnh if else**

|  |  |  |
| --- | --- | --- |
| If (condition) {           //Câu lệnh code  } | If (condition) {           Code 1;  }  Else {           Code 2;  } | If (condition1) {           Code 1;  }  Else if (condition2) {           Code 2;  }  ….  Else {           Code n;  } |
| Nếu condition đúng thì thực hiện code | Nếu condition đúng thì thực hiện code 1 ngược lại thực hiện code 2 | Kiểm tra lần lượt từ condition từ trên xuống và thực hiện code tương ứng |

**Câu lệnh switch case**

Switch (biến) {

         Case giá trị 1: {

                     Code 1;

                     Break;

}

Case giá trị 2: {

         Code 2;

         Break;

}

         …

Default: {

         Code n;

         Break;

}

}

**VÒNG LẬP FOR**

For (khởi tạo giá trị cho biến lập, điều kiện lập, bước nhảy) {

//code;

}

*Ví dụ*

for (int i=0; i<n; i++) {

//code;

}

Thực hiện code n lần (tương ứng với I chạy từ 0 -> n)

Quy trình chạy

Chạy vòng for đầu tiên với giá trị biến lặp bằng giá trị khởi tạo

Sau đó thực hiện bước nhảy và kiểm tra điều kiện

Luôn luôn lặp 1 vòng lặp đầu tiên.

**Break: gặp câu lệnh này sẽ dừng ngay vòng lặp gần nhất**

**Continute: gặp câu lệnh này thì bỏ qua tất cả câu lệnh phía dưới câu lệnh này và quay lên thực hiện một vòng lặp mới.**

**VÒNG LẶP WHILE**

While (condition) {

//code;

}

Thực hiện code đến khi nào condition còn đúng.

*Tách để thao tác trên các chữ số của một số*

While (n! =0) {

Int t=n%10;

//Thao tác trên t;

n/=10;

}

*Đếm số chữ số của n*

Int d=0;

While (n! =0) {

d++;

n/=10;

}

*Tính tổng chữ số của n*

Int s=0;

While (n) {

S+=n%10;

n/=10;

}

**VÒNG LẶP DO WHILE**

Do {

Code;

} while (condition);

Luôn thực hiện 1 vòng lặp đầu tiên sau đó check condition

**HÀM TRONG C/C++**

***Kiểu dữ liệu hàm tên\_hàm (các tham số, biến cần truyền vào) {***

***Câu lệnh trong hàm.***

***}***

**Một số kiểu dữ liệu hàm**

Void: Không có giá trị trả về

Các kiểu dữ liệu khác (int, long long, float, string, char…) có kiểu dữ liệu trả về.

Return giatri (tương ứng với kiểu dữ liệu hàm)

Ví dụ

Void tenham (kieu du lieu a, kieu du lieu b, …) {

}

Trong hàm main

Tenham (giá trị 1, giá trị 2…)

Trong đó

a, b, … là tham số.

giá trị 1, giá trị 2, … là đối số

Mỗi khi thực hiện hàm thì sẽ lấy gán giá trị a bằng giá trị đối số tương ứng mà không làm thay đổi giá trị đối số sau khi hàm thực hiện xong.

Lưu ý kiểu dữ liệu giữa tham số và đối số tương ứng phải giống nhau.

Lưu ý

|  |
| --- |
| Lập trình C |
| Nếu mún thay đổi giá trị của đối số sau khi hàm kết thúc thì ta thực hiện truyền con trỏ  Kieu\_du\_lieu\_ham tenham (kieudulieu \*tenbien, kieudulieu \*tenbien, …) {  } |

**MẢNG MỘT CHIỀU TRONG C++/C**

**1.Khai báo**

Kieu\_du\_lieu tenmang [số lượng phần tử của mảng]

Ví dụ:

Int a [100];

Long long b [100];

Hoặc int mang [6] = {1,2,3,4,5,6}; //Gán giá trị cho từng mảng

**2.Cách hoạt động của mảng**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Phần tử | 1 | 2 | 3 | 4 | 5 | 6 | 7 | .. | n |
| Chỉ số | 0 | 1 | 2 | 3 | 4 | 5 | 6 | .. | n-1 |

**3.Truy cập đến phần tử trong mảng**

Tên mảng [chỉ số] với chỉ số chạy từ 0->n-1;

**4.Nhập mảng**

|  |
| --- |
| For (int i=0; i<n; i++) {  Scanf (“%d”, &a[i]);  } |

**5.Duyệt mảng**

|  |
| --- |
| For (int i=0; i<n; i++) {  Thao tác trên a[i]  } |

**MẢNG HAI CHIỀU**

**1.Khai báo**

Kieudulieu tenmang [số lượng dòng] [số lượng cột];

Int a [2] [2] = {{1,2,3}, {4,5,6}}

**2.Cách thức tổ chức**

Ví dụ int a [2] [3]

|  |  |  |  |
| --- | --- | --- | --- |
|  | 0 | 1 | 2 |
| 0 |  |  |  |
| 1 |  |  |  |

Trong bộ nhớ máy tính là lưu trữ thành 1 dòng các phần tử liền kề nhau

**3.Truy cập các phần tử**

Tenmang [chỉ số hàng] [chỉ số cột];

**4.Duyệt qua tất cả các phần tử**

For (int i=0; i<n; i++) {

For (int j=0; j<m; j++) {

Thao tác trên a [i] [j]

}

}

Lưu ý khi truyền mảng 2 chiều vào hàm phải có 1 trong 2 thông số cụ thể hàng và cột

**MẢNG KÍ TỰ CHAR [] TRONG C++**

**1.Khai báo**

Char ten\_xau [x];

Với x là số lượng kí tự tối đa trong xâu

Các thư viện

#include <cstdlib>

#include <cctype>

#include <string.h>

**2.Nhập xâu**

Hàm gets(ten\_xâu): Dùng dê nhập xâu có dấu cách (C)

Scanf (“%s”, tenxau): Dùng để nhâp xâu không dấu cách (C)

Hàm fgets (ten chuỗi kí tự, số lượng tối đa, stdin);

Cin>>ten\_xau: Dùng để nhập xâu không có dấu cách (c++)

Cin.getline(ten\_xau): Dùng để nhập xâu có dấu cách. (c++)

*Lưu ý: Trước gets có xuất hiện cin thì cần có getchar ()*

**3.Một số hàm trong mảng char và xử lý kí tự**

***Hàm strlen(tên\_xâu): Số lượng kí tự của mảng char***

|  |  |
| --- | --- |
| Kiểm tra in thường | Trả về giá trị 0 -> Sai |
| Trả về giá trị khác 0 -> Đúng |

***Hàm isupper (char c): Kiểm tra in hoa***

|  |  |
| --- | --- |
| Kiểm tra in hoa | Trả về giá trị 0 -> Sai |
| Trả về giá trị khác 0 -> Đúng |

***Hàm isalpha (char c): Kiểm tra kí tự chữ cái***

|  |  |
| --- | --- |
| Kiểm tra kí tự chữ cái | Trả về giá trị 0 -> Sai |
| Trả về giá trị khác 0 -> Đúng |

***Hàm isdigit (char c): Kiểm tra ki tự số***

|  |  |
| --- | --- |
| Kiểm tra kí tự số | Trả về giá trị 0 -> Sai |
| Trả về giá trị khác 0 -> Đúng |

***Hàm to\_upper (char c): Biến kí tự thành in hoa***

Biến kí tự thành chữ in hoa

Char c= to\_upper(c);

***Hàm to\_lowwer (char c): Biến kí tự thành in thường***

Biến kí tự thành in thường

Char c= to\_lowwer(c);

***Chuyển kí tự về thành số***

C[i] – ‘0’

Hoặc c[i] – 48;

*Lưu ý kí tự cuối cùng của mỗi mảng kí tự là kí tự NULL*

***Hàm strcpy (char c [], char d [])***

Copy xâu d gán vào xâu c

***Hàm strcat (char c [], char d [])***

Nối xâu d vào xâu c

***Hàm strcmp (char c [], char d []):***

|  |  |
| --- | --- |
| Strcmp (char c [], char d []) | Trả về 0 nếu c=d |
| Trả về 1 nếu c>d |
| Trả về -1 nếu c<d |

Dùng để so sánh hai xâu theo thứ tự từ điển

***Hàm Strrev (char c []) Đảo ngược xâu***

Char [100] c= Strrev (d);

***Hàm strstr (char c [], char d []): Kiểm tra xâu c có chứa xâu d không***

|  |  |
| --- | --- |
| Kiểm tra xâu d trong xâu c | Nếu tìm thấy con trỏ trả về vị trí bắt đầu xâu d trong xâu c |
| Nếu không tìm thấy trả về NULL |

***Tách từ bằng strtok***

Strtok (chuỗi cần tách, “kí tự mà bạn mún tách”)

Char \*ten\_con\_tro = strtok (ten\_xau, “kí tự”);

While (tencontro! =NULL) {

//Con trỏ lưu từng từ

//Thao tác trên con trỏ.

Tencontro=strtok (NULL,” “);

}

***Tách từ lưu vào mảng hai chiều***

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| i/j | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | … |
| 0 | L | E |  |  |  |  |  |  |  |
| 1 | T | U | A | N |  |  |  |  |  |
| 2 | B | I | N | H |  |  |  |  |  |
| 3 | 2 | 7 |  |  |  |  |  |  |  |
| 4 | 0 | 8 |  |  |  |  |  |  |  |
| 5 | 2 | 0 | 0 | 3 |  |  |  |  |  |
| …. |  |  |  |  |  |  |  |  |  |

a[0] = “LE”

a[0][1] =’E’;

int n=0;

char \*t=strtok (c,” “);

while (t! = NULL) {

strcpy (a[n], t);

n++;

t=strtok (NULL, “ “);

}

***Mã ascii***

|  |  |
| --- | --- |
| 0 -> 9 | 48 -> 57 |
| a -> z | 97 -> 122 |
| A -> Z | 65 -> 90 |

***Bài toán dếm tần xuất kí tự***

Cnt [256] = {0};

For (int i=0; i<strlen(c); i++) {

Cnt[c[i]] ++;

}

***Mảng kí tự char động trong C***

Char \*ten\_chuoi;

String = array of chars + ‘\0’;

Phần tử cuối cùng sau mỗi chuỗi đều kết thúc bằng 1 kí tự NULL hoặc ‘\0’

Lưu chuỗi n kí tự cần một chuỗi có n+1 phần tử.

Cấp phát bộ nhớ khi đã biết kích thước

Char \*ten\_chuoi = new char [số lượng phần tử];

Hàm strcpy: dùng để sao chép chuỗi

**strcpy (s1, s2);** //Copy nội dung xâu s2 sang s1.

**strlen(s1):** //Trả về độ dài của chuỗi s1;

Gán giá trị cho một con trỏ char

Char \*s1 = “Hello”;

Char \*s2= new char [strlen (s1) + 1];

strcpy (s2, s1);

**strdup (s1)** //trả về giá trị xâu copy từ xâu s1

char \*s2=n strdup (s1); //s2 mang giá trị xâu s1

Lưu ý: Sau khi cấp phát phải có lệnh **free (ten con tro duoc cap phat);**

**strcmp (s1, s2):** So sánh xâu s1 và s2;

Trả về 0 nếu nếu s1 = s2.

Trả về 1 nếu s1 lớn hơn s2.

Trả về -1 nếu s1 bé hơn s2.

**strcat (s1, s2);** //Nối chuỗi s2 vào chuỗi s1.

**strstr (s1, s2)** //Kiểm tra xâu s2 trong xâu s1

Trả về địa chỉ của kí tự đầu tiên của s2 trong s1. Nếu không tìm thấy thì trả về con trỏ NULL.

**Nhập chuỗi động**

Muốn nhập một chuỗi động ta cần mượn một chuỗi tĩnh để nhập sau đó cấp phát bộ nhớ cho chuỗi động để sao chép bằng các hàm strcpy hay strdub.

Ví dụ

Char tmp [50];

Cin.getline (tmp);

Char \*s= new char [strlen(tmp) +1];

Strcpy (s, tmp);