NHẬP MÔN LẬP TRÌNH

CHƯƠNG 1. TỔNG QUAN

Mục tiêu

- Tổng quan về lập trình
- Làm quen với NNLT C.
- Biết cấu trúc 1 chương trình viết bằng NNLT C.
- Biết các thư viện chuẩn
- Hiểu và vận dụng các kiểu dữ liệu cơ bản, hằng, biến của NNLT C.

Nội dung

1. Tổng quan về lập trình
2. Giới thiệu về ngôn ngữ lập trình C
3. Đặc điểm của ngôn ngữ lập trình C
4. Cấu trúc chương trình C
5. Thư viện hàm chuẩn C
6. Ưu và nhược
7. Bài tập

1. Tổng quan về lập trình

- Một chương trình (program) là một dãy các chỉ thị (instruction) điều khiển sự hoạt động của máy tính nhằm giải quyết một công việc nào đó.
- Người viết chương trình (còn gọi là lập trình viên) là những người tạo lập ra các chương trình điều khiển máy tính.

1. Tổng quan về lập trình

- Ngôn ngữ lập trình là ngôn ngữ được lập trình viên sử dụng để viết chương trình cho máy tính.
- Khi một chương trình được viết bằng một NNLT nào đó thì các chỉ thị, câu lệnh trong chương trình phải tuân theo các quy tắc, các luật do NNLT đó quy định.
- Người lập trình thường viết chương trình bằng các NNLT cấp cao vì tính dễ dùng, có thể diễn đạt được các ý tưởng trừu tượng và có tính tương thích cao.
- Một số NNLT thông dụng: C/C++, C#, Java, Python,…

2. Giới thiệu về ngôn ngữ lập trình C

- C là ngôn ngữ lập trình cấp cao, được sử dụng rất phổ biến để lập trình hệ thống cùng với Assembler và phát triển các ứng dụng.
- Ngôn ngữ lập trình C là một ngôn ngữ lập trình hệ thống rất mạnh và rất "mềm dẻo", có một thư viện gồm rất nhiều các hàm (function) đã được tạo sẵn.
- Hỗ trợ rất nhiều phép toán nên phù hợp cho việc giải quyết các bài toán kỹ thuật có nhiều công thức phức tạp.
- Cho phép người lập trình tự định nghĩa thêm các kiểu dữ liệu trừu tượng mới.

3. Đặc điểm của ngôn ngữ lập trình C

- Tính cô đọng (compact)
- Tính cấu trúc (structured)
- Biên dịch (compile)
- Tính linh động (flexible)
- Tính tương thích (compatible)

4. Cấu trúc chương trình C

Một chương trình C bao gồm các phần như: Các chỉ thị tiền xử lý, định nghĩa kiểu dữ liệu mới, khai báo biến ngoài, các hàm tự tạo, hàm main.

Cấu trúc:

1. *Các chỉ thị tiền xử lý*
2. *Định nghĩa kiểu dữ liệu*
3. *Khai báo các biến ngoài*
4. *Khai báo các prototype của hàm tự tạo*
5. *Hàm main*
6. *Định nghĩa các hàm tự tạo*

4. Cấu trúc chương trình C

Các chỉ thị tiền xử lý:

#include <Tên tập tin thư viện>

#define ….

Chỉ thị #define được sử dụng trong việc định nghĩa các ký hiệu

Ví dụ:

```
#include <stdio.h>
#define  MAX  100
#define  SIZE  25
```

4. Cấu trúc chương trình C

Định nghĩa kiểu dữ liệu (không bắt buộc): dùng để đặt tên lại cho một kiểu dữ liệu nào đó để gợi nhớ hay đặt 1 kiểu dữ liệu cho riêng mình dựa trên các kiểu dữ liệu đã có.

Cú pháp: **typedef** <Tên kiểu cũ> <Tên kiểu mới>

***Ví dụ*:**

typedef int SoNguyen; // Kiểu SoNguyen là kiểu int

4. Cấu trúc chương trình C

Khai báo các biến ngoài:

Được sử dụng để khai báo các biến toàn cục khi cần thiết.
Phần này không bắt buộc khai báo trong chương trình.

Cú pháp: <Kiểu dữ liệu> <Tên biến>;

Ví dụ: int a; //khai báo biến số nguyên a

Khai báo các Prototype của hàm tự tạo:

Cú pháp: <Kiểu trả về của hàm> <Tên hàm>([<các đối số>]);

Ví dụ: boolean isPrime(int a); // prototype của hàm isPrime

4. Cấu trúc chương trình C

Hàm main:

Khi chương trình thực thi thì hàm Main được gọi trước tiên. Đây là phần bắt buộc khai báo trong chương trình.

Cú pháp:

```
<Kiểu dữ liệu trả về> main()
{
  [//các khai báo cục bộ trong hàm ]
  [//Các câu lệnh dùng để định nghĩa hàm main]
  [return <kết quả trả về>; ]
}
```

Ví dụ:

```
void main()
{
        printf("Hello");
        getch();
}
```

4. Cấu trúc chương trình C

Định nghĩa các hàm tự tạo:

Đây là phần không bắt buộc trong chương trình

Cú pháp: <Kiểu dữ liệu trả về> <Tên hàm>([<Các đối số>])

```
{
        [//các khai báo cục bộ trong hàm ]
        [//Các câu lệnh dùng để định nghĩa hàm]
        [return <kết quả trả về>; ]
}
```

Ví dụ: Viết hàm trả về giá trị tổng hai số nguyên a và b.

```
int Tinhtong(int a, int b)
{
        int c = a + b;
        return c;
}
```

4. Thư viện hàm chuẩn C

Tất cả trình biên dịch C đều chứa thư viện hàm chuẩn C.

Một số thư viện chuẩn trong C:

- **stdio.h:** Tập tin chứa các hàm vào/ra chuẩn.
- **conio.h**: Tập tin định nghĩa các hàm vào ra trong chế độ DOS (DOS console).
- **math.h**: Tập tin định nghĩa các hàm tính toán.
- **alloc.h**: Tập tin định nghĩa các hàm liên quan đến việc quản lý bộ nhớ.
- **io.h**: Tập tin định nghĩa các hàm vào ra cấp thấp.
- **graphics.h:** Tập tin định nghĩa các hàm liên quan đến đồ họa.

5. Các hàm nhập xuất

- Hàm printf() và scanf():
  - Hàm printf()

Cú pháp :

#include <stdio.h>

printf(<format> [,<arguments>, ...]);

<format>: xâu định dạng

<arguments>: các tham số tương ứng 2.

Công dụng : Xuất chuỗi thông báo hoặc các giá trị tham số hay vừa chuỗi thông báo kết hợp các giá trị tham số ra màn hình.

5. Các hàm nhập xuất

- Hàm printf()

Định dạng Kiểu Ghi chú

%d int Số nguyên

%E float, double Dấu chấm động

%f, %lf float, double Dấu phẩy tĩnh

%c char Ký tự

%s char * Xâu ký tự

5. Các hàm nhập xuất

- Hàm printf()

Ví dụ :

```
printf("n = %d\n", -10); /*  -10 */
printf("A : %c\n", 'A'); /* --> A */
printf("A : %d\n", 'A'); /* A: 65 */
printf("f = %.2f", 123.4); /* ? */
```

5. Các hàm nhập xuất

- Hàm scanf ()

Cú pháp :

#include <stdio.h>

scanf(<format>, <&tên biến>);

<format>: xâu định dạng

<&tên biến>: địa chỉ của các tham số tương ứng

Công dụng : Nhập dữ liệu từ bàn phím và gán giá trị vào địa chỉ của các tham số tương ứng.

5. Các hàm nhập xuất

- Hàm scanf ()

Ví dụ:

```
int n; float a; char c;
printf("Nhap so nguyen:");
scanf("%d", &n);
printf("Nhap so thuc:");
scanf("%d", &a);
printf("Nhap ky tu:");
fflush(stdin);
scanf("%c", &c);
```

- Ví dụ

```
int a, b, c;
printf("Enter the first value:");
scanf("%d", &a);
printf("Enter the second value:");
scanf("%d", &b);
c = a + b;
printf("%d + %d = %d\n", a, b, c);
```

Hàm getch() và getche()

- Cú pháp :

```
#include <conio.h>
int getche();
```

Công dụng : Dùng để đọc một ký tự từ bàn phím và trả về ký tự đó, đồng thời ký tự nhập sẽ xuất hiện trên màn hình tại vị trí con cursor. Thường dung để dừng chương trình để xem kết quả

Hàm getch() và getche()

- Cú pháp :

```
#include <conio.h>
int getch();
```

Công dụng : Dùng để đọc một ký tự từ bàn phím và trả về ký tự đó, đồng thời ký tự nhập sẽ **không** xuất hiện trên màn hình tại vị trí con cursor. Thường dung để dừng chương trình để xem kết quả

Nhập xuất dung cin và cout

Xuất dữ liệu bằng cout:

Cú pháp : #include <iostream>

using namespace std;

cout<<expression;

expression : các tham số, chuỗi, biểu thức

Công dụng : Dùng để xuất chuỗi, các giá trị biến hay vừa chuỗi và giá trị biến ra màn hình.

Ví dụ:

cout<<“a = “<<a<<endl;

cout<<“\tKet qua cua chuong trinh la:”<<kq<<“\n”;

Nhập xuất dung cin và cout

- Nhập dữ liệu bằng cin:

  Cú pháp : #include <iostream>

  using namespace std;

  cin>>variable;

  variable : các biến

Công dụng : Dùng để nhập dữ liệu các giá trị biến vào địa chỉ biến.

```
#include<conio.h>
#include <iostream>
using namespace std;
void main(void)
{
        string name; int age; float weight;
        cout<<“You Name :”;
        cin>>name;
        cout<<“You age :”;
        cin>>age;
        cout<<“You weight :”;
        cin>>weight;
        cout<<endl<<“Ten : ”<<name<<“\tTuoi : “ <<age;
        cout<<“\t trong luong :”<<weight<<endl;
        getch();
}
```

Bài tập

1. Viết chương trình xuất ra câu thông báo: “Chao ban den voi ngon ngu C”.
2. Viết chương trình xuất ra đoạn thông báo:

   “Chao ban!

   Day la chuong trinh C dau tien.

   Vui long nhan phim Enter de ket thuc.”
3. Viết chương trình nhập vào 1 số nguyên, xuất ra màn hình số nguyên vừa nhập.
4. Viết chương trình nhập vào 2 số nguyên, tính và xuất kết quả tổng, hiệu, tích và thương của 2 số nguyên vừa nhập.
5. Viết chương trình xuất ra các số từ 1 đến 10.


STOP
BLUEBUDDIES