Nội dung viết dưới đây không phải quy định, tuy nhiên các bạn nên tuân theo để nhằm mục đích thống nhất chung giữa các thành viên trong nhóm, giữa cố vấn hướng dẫn với các bạn trong nhóm. Tài liệu được tham khảo từ nhiều nguồn khác nhau, những nội dung được trình bày dưới đây đã từng được nhiều lập trình viên chuyên nghiệp sử dụng.

1. **Khoảng trống (dấu cách):**

   Sử dụng dấu cách để phân biệt rõ ràng giữa các phép tính toán, khai báo.

   Đặt dấu cách giữa từ khóa #include và tên thư viện:

   ```
   #include <stdio.h>
   ```

   Nên sử dụng dấu "<" và ">" để khai báo các thư viện chuẩn của ngôn ngữ C.

   Không đặt các dấu cách ở cuối các dòng lệnh, không nên đặt liên tiếp nhiều dấu cách với nhau.

   Sau dấu "(" và **trước** dấu ")" không nên sử dụng dấu cách. Tuy nhiên **trước** dấu "(" và **sau** dấu ")" có thể đặt thêm dấu cách linh động cho phù hợp.

   Khi khai báo nhiều biến có liên quan với nhau và cùng kiểu dữ liệu liên tiếp, sử dụng dấu cách sau dấu "," phân cách.

   Đặt dấu cách giữa các toán tử thực hiện phép tính, Ví dụ:

   ```
   a = a + b;
   b = b + c;
   d += c;
   if (a >= b)
   if (a % b)
   for (int i = 0; i < n; i++)
   ```

   Viết theo chuẩn này

   ```
   a = a +b;
   b= b+c;
   d +=c;
   if (a>=b)
   if (a% b)
   for (int i=0; i<n; i++)
   ```

   Không viết theo cách này

2. **Dấu tab**

   Sử dụng dấu tab để lùi dầu dòng nằm trong một khối lệnh hoặc một lệnh thực hiện bên trong lệnh khác.

   Ví dụ:

   ```
   #include <stdio.h>

   main() {
       //Lùi tab đến đây.
       int a, b, c, d, n;
       a = a + b;
       b = b + c;
       d += c;
       if (a >= b) {
           //Bắt đầu viết lệnh từ đây.
       }
       if (a > b && b > c) {
           //Viết lệnh từ đây.
       }
       for (int i = 0; i < n; i++) {
           //Viết lệnh từ đây.
       }
   }
   ```

3. Cách đặt tên

Các tên được đặt nên mang một ý nghĩa nhất định. Không sử dụng dấu cách để đặt tên

Nên đặt tên bằng tiếng Anh.

Tên các biến, các hàm bắt đầu bằng chữ cái in thường, ví dụ: length, height, size, …

Nếu tên biến hoặc hàm có chứa 2 từ trở lên thì chữ cái đầu tiên của mỗi từ kể từ từ thứ 2 nên viết hoa, ví dụ: myName, mySchoolName, getSum, getText, setSize, …

Sử dụng dãy chữ cái in hoa toàn bộ để đặt tên cho hằng số, dùng dấu gạch dưới để nối các từ, ví dụ: PI, MAX, MY_COUNTRY, …

Các biến đặt cho số thường dùng: a, b, c, d, …

Các biến đặt cho số lượng: m, n, …

Các biến kiểu xâu: s, s1, s2, …

Các biến chạy trong các vòng lặp, sử dụng cho chỉ số: i, j, k, …

Tên cho struct được viết hoa chữ cái đầu mỗi từ, ví dụ: SinhVien, ConMeo, …

Ví dụ:

```
// Hằng Số
#define MY_SCHOOL "PTIT"
// Tên biến
int age;
int height, weight;
char myName[30], myTeacherName[30];
// Tên hàm
int sum(int a[], int n) {
    //
}
float getAverage(int a[], int n) {
    //
}
```

```
// Khai báo cấu trúc
typedef struct SinhVien{
        int mssv;
        char hoTen[20];
        float diemTrungBinh;
} SinhVien;
```

4. Sử dụng { }

Dấu { } để đánh dấu bắt đầu và kết thúc cho một khối lệnh.

Nên đặt dấu "{" cùng dòng với các tên hàm, các điều kiện của if, while đứng trước nó, đặt dấu cách trước "{". Không nên đặt đấu "{" ở dòng tiếp theo.

Dấu "}" phải được đặt thẳng hàng với chữ cái đầu tiên của dòng chứa dấu "{" tương ứng.

Nên sử dụng khối lệnh trong { } ngay cả khi chỉ có một câu lệnh sau if, else, while …, đây là một thói quen tốt.

Ví dụ:

```
main() {
    int a, b, c, d, n;
    if (a >= b) { // Đặt dấu { ở đây
        // Câu lệnh
    }
    if (a > b && b > c)
    { // Không nên đặt dấu { ở đây.
        // Câu lệnh
    }
    for (int i = 0; i < n; i++) {
        // Câu lệnh
    } // Dấu } phải đặt thẳng hàng với for
} // Dấu } phải đặt thẳng hàng với main
```

5. **Chú thích**

Chú thích những nơi cần thiết, nội dung chú thích trong chương trình đơn giản, dễ hiểu để tiện cho việc đọc lại code. Nên sử dụng chú thích theo dòng bằng cách thêm dấu // vào đầu dòng thay vì sử dụng /* */.

Không nên chú thích những vẫn đề đã quá rõ ràng.

Sử dụng chú thích để tạm thời vô hiệu hóa đoạn chương trình để test chương trình.

Mục đích của việc tuân theo các quy chuẩn này là để giúp cho bạn hoặc các lập trình viên khác dễ đọc lại mã nguồn, dễ dàng sửa lỗi, dễ dàng nâng cấp. Các bạn có thể sử dụng một cách linh động. Được phép vi phạm các quy tắc này trong các trường hợp cần thiết. Đặc biệt trong quá trình học tập nếu giảng viên yêu cầu cứng nhắc trong việc sử dụng phong cách lập trình cá nhân thì cần sử dụng cho riêng môn học đó.

Các ý kiến đóng góp xin gửi về admin@nvtien.info.