Bài thực hành tuần 5

1. Nội dung

- Mảng: khai báo, truy xuất mảng
- Mảng tĩnh và mảng sử dụng thư viện <vector>
- Duyệt mảng
- Hàm có tham số là mảng

2. Khởi tạo và in mảng

Sinh viên tập viết các đoạn mã thao tác với mảng dưới đây. Cách thực hành như sau:

- Chép lại đoạn mã
- Chạy thử, nắm bắt các công việc mà đoạn mã thực hiện
- Tạo file mới và đánh lại đoạn mã theo trí nhớ và cách hiểu của mình

1. Khai báo mảng

Khai báo mảng tĩnh

```
int main()
{
    const int N = 10;
    int num[N]; // mảng num có 10 số nguyên

    const int N_str = 100;
    string str[N_str]; // mảng str có 100 xâu kí tự

    const int N_double = 1000;
    double real[N_double]; //mảng real có 1000 số thực

    return 0;
}
```

2. Khởi tạo mảng bằng vòng lặp

Khởi tạo mảng bằng vòng lặp

```
    for (int i = 0; i < N; i++) {
        num[i] = 0; // khởi tạo bằng số 0
    }

    for (int i = 0; i < N_str; i++) {
        str[i] = "Hello"; // khởi tạo với xâu "Hello"
    }

    for (int i = 0; i < N_double; i++) {
        real[i] = 0.5; // khởi tạo với số 0.5
    }
```

3. In mảng bằng vòng lặp

In giá trị trong mảng

```
for (int i = 0; i < N; i++) {
    cout << num[i] << endl;
}

for (int i = 0; i < N_str; i++) {
    cout << str[i] << endl;
}

for (int i = 0; i < N_double; i++) {
    cout << real[i] << endl;
}
```

Với các bài tập dưới đây, hãy sử dụng các đoạn mã in giá trị trong mảng tương tự như trên để thấy kết quả của các thao tác trên mảng

4. Khởi tạo mảng với các giá trị liên tiếp

Khởi tạo mảng với các giá trị liên tiếp

```
int start = 2;
for (int i = 0; i < N; i++) {
    num[i] = start+i;
}
```

5. Khởi tạo mảng ngẫu nhiên

Khởi tạo mảng với các giá trị ngẫu nhiên

```
for (int i = 0; i < N_double; i++) {
    real[i] = 1.0 * rand() / RAND_MAX;
}
```

3. Tính toán trên mảng

1. Tính tổng các phần tử trong mảng

Tính tổng các phần tử trong mảng

```
int sum = 0;
for (int i = 0; i < N; i++) {
    sum += num[i];
}
cout << sum << endl;
```

2. Tính tổng các phần tử có lựa chọn

Tính tổng các phần tử chia ba dư 1

```
int sum = 0;
for (int i = 0; i < N; i++) {
    if (num[i] % 3 == 1) {
        sum += num[i];
    }
}
cout << sum << endl;
```

3. Tính các số Fibonacci đầu tiên

Tính các số Fibonacci đầu tiên

```
num[0] = 1;
num[1] = 1;
for (int i = 2; i < N; i++) {
    num[i] = num[i-1] + num[i-2];
}
```

4. Tính các từ “Fibonacci” đầu tiên (cộng xâu)

Tính các số Fibonacci đầu tiên

```
str[0] = "a";
str[1] = "b";
for (int i = 2; i < N; i++) {
    str[i] = str[i-1] + str[i-2];
}
```

5. Tích vô hướng của hai vec-tơ

Tính các số Fibonacci đầu tiên

```
double v1[N], v2[N];
for (int i = 0; i < N; i++) {
    v1[i] = 1.0 * rand() / RAND_MAX;
    v2[i] = 1.0 * rand() / RAND_MAX;
}

double dot = 0;
for (int i = 0; i < N; i++) {
    dot += v1[i] * v2[i];
}
cout << dot << endl;
```

4. Một số thao tác đơn giản trên mảng

1. Nhập mảng

Nhập mảng

```
for (int i = 0; i < N_str; i++) {
    cin >> str[i];
}
```

2. Sao chép mảng

Sao chép mảng

```
string strCopy[N_str];
for (int i = 0; i < N_str; i++) {
    strCopy[i] = str[i];
}
```

3. Duyệt mảng để tìm kiếm

Tìm kiếm vị trí và giá trị trong mảng thỏa mãn điều kiện

```
for (int i = 0; i < N_double; i++) {
```

```
        if (real[i] < 0.5) {  // tìm các số nhỏ hơn 0.5
            cout << i << " " << real[i] << endl;
        }
    }
```

4. Mảng động với <vector>

Để sử dụng mảng động với thư viện <vector>, bạn cần sử dụng câu lệnh #include <vector>. Sau đó sử dụng khai báo

vector< tên_kiểu > tên_mảng (số_phần_tử, giá_trị_khởi_tạo);

Sau khai báo này, bạn có thể sử dụng, truy xuất mảng giống hệt như khai báo mảng tĩnh. Tất nhiên, sử dụng <vector> còn nhiều điểm lợi khác, chúng ta sẽ thực hành trong buổi sau.

5. Hàm có đối số là mảng

Hãy viết các hàm sau và sử dụng nó trong hàm main()

1. Viết hàm in mảng có N số nguyên

Hàm in các giá trị trong mảng

```
void printArray(int arr[], int N)
{
    for (int i = 0; i < N; i++) {
        cout << arr[i] << endl;
    }
}
```

2. Viết hàm nhập một mảng có N xâu ký tự (tự viết).

3. Viết hàm khởi tạo mảng có N số nguyên bằng giá trị cho trước.

Hàm khởi tạo mảng với giá trị cho trước

```
void initArray(int arr[], int N, int value)
{
    for (int i = 0; i < N; i++) {
        arr[i] = value;
    }
}
```

4. Viết hàm khởi tạo mảng có N số nguyên liên tiếp bắt đầu từ một giá trị cho trước

Hàm khởi tạo mảng với giá trị liên tiếp

```
void initArray(int arr[], int N, int value)
{
    for (int i = 0; i < N; i++) {
        arr[i] = value + i;
    }
}
```

5. Viết hàm khởi tạo mảng có N số thực ngẫu nhiên

Hàm khởi tạo mảng ngẫu nhiên trong khoảng [low, high]

```
void initArrayRandom(double arr[], int N, double low, double high)
{
    for (int i = 0; i < N; i++) {
        arr[i] = (high-low) * rand() / RAND_MAX + low;
    }
}
```

6. (Bài tập) Viết hàm tính tổng của mảng có N số nguyên
7. (Bài tập) Viết hàm nối tất cả các xâu trong mảng có N xâu ký tự, có thể dùng 1 xâu làm dấu phân cách.
8. (Bài tập) Viết hàm tính tích vô hướng của 2 mảng có N số thực
9. (Bài tập) Sử dụng <vector> cho các hàm trên.