Bài thực hành tuần 6

1. Nội dung

Các thao tác với mảng và xâu

- Sửa các phần tử trong mảng
- Chèn thêm phần tử vào mảng
- Xóa phần tử khỏi mảng
- Sắp xếp các phần tử trong mảng
- Thao tác với xâu

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

- Chép lại đoạn mã hàm
- Viết đoạn mã chạy thử hàm vừa viết
- Tạo file mới và đánh lại đoạn mã theo trí nhớ và cách hiểu của mình

2. Sửa các phần tử trong mảng

1. Dùng vòng lặp xuôi sửa các phần tử lớn hơn 4 thành 4

Hàm sửa các phần tử lớn hơn 4 - Vòng lặp xuôi

```
void convertBiggerThan4(int arr[], int n)
{
    for (int i = 0; i < n; i++) {
        if (arr[i] > 4) {
            arr[i] = 4;
        }
    }
}
```

Để sử dụng hàm trên, viết trong hàm **main()** đoạn mã sau (tự viết hàm **printArray()**)

Sử dụng hàm convertBiggerThan4

```
    int n = 7;
    int a[100] = { 2, 5, 6, 2, 3, 3, 5 };

    printArray(a, n);

    convertBiggerThan4(a,n);
    printArray(a, n);
```

2. Dùng vòng lặp ngược sửa các phần tử thành **tổng hai phần tử phía trước**

Sửa các phần tử thành tổng hai phần tử phía trước – Vòng lặp ngược

```
void convertSumTwoPreviousNumber(int arr[],int n){
    for (int i = n-1; i >= 0; i--)
    {
        arr[i] = ( (i > 0) ? arr[i-1] : 0 )
         + ( (i > 1) ? arr[i-2] : 0 );
    }
```

```
}
```

Ta cần sử dụng vòng lặp ngược để tránh các phần tử của mảng bị thay đổi trước khi làm phép cộng. Hơn nữa, ở đây ta dùng toán tử lựa chọn **? :** để đảm bảo chỉ số mảng hợp lệ khi truy xuất mảng.

3. Sửa các phần tử thành tổng hai phần tử bên cạnh bằng cách dùng một mảng khác sao chép các phần tử của mảng trước khi làm phép toán

Sửa các phần tử thành tổng hai phần tử bên cạnh – Sao chép mảng

```
void convertWithNextToNumbers(int arr[], int n)
{
    int temp[1000];
    for (int i = 0; i < n; i++) {
        temp[i] = arr[i];
    }

    for (int i = 0; i < n; i++) {
        arr[i] = (i > 0 ? temp[i-1] : 0) +
           (i < n-1 ? temp[i+1] : 0);
    }
}
```

4. Sửa các phần tử thành tổng hai phần tử bên cạnh bằng cách "khéo hơn"dùng biến lưu trữ các phần tử của mảng trước khi làm phép toán

Sửa các phần tử thành tổng hai phần tử bên cạnh – Không cần sao chép mảng

```
void convertWithNextToNumbersWithoutCopy(int a[], int n)
{
    int prev = 0;
    for (int i = 0; i < n; i++) {
        int real_prev = prev;
        prev = a[i];
        a[i] = ( i > 0 ? real_prev : 0 ) +
               ( i < n-1 ? a[i+1] : 0 );
    }
}
```

5. (Bài tập) Sửa các phần tử thành tổng tất cả các phần tử phía trước và chính nó.
6. (Bài tập) Sửa các phần tử thành số nhỏ nhất trong tất cả các phần tử phía trước và chính nó.
7. (Bài tập) Sửa các phần tử thành trung bình cộng của k số tính từ phần tử đó về phía trước (k > 0).

3. Chèn thêm phần tử vào mảng

1. Chèn thêm một phần tử vào một vị trí trong mảng

Chèn phần tử vào mảng

```
int insert(int arr[], int n, int pos, int value)
{
    for (int i = n; i > pos; i--) {
        arr[i] = arr[i-1];
```

```
    }
    arr[pos] = value;
    return n+1;
}
```

Ở đây, hàm trả về số phần tử mới của mảng (tăng 1).

2. (Bài tập) Thêm đoạn mã đảm bảo giá trị **pos** hợp lệ trước khi thêm phần tử
3. Chèn một mảng vào một vị trí trên mảng khác

Chèn một mảng vào một mảng khác

```
int insertArrayToArray(int a[], int b[], int na, int nb, int pos)
{
    for (int i=na+nb-1; i>=nb+pos; i--) a[i]=a[i-nb];
    for (int i=0; i<nb; i++) a[i+pos]=b[i];
    return na+nb;
}
```

Hàm trả về số phần tử mới của mảng **a**.

4. (Bài tập) Thêm đoạn mã đảm bảo giá trị **pos** hợp lệ.

4. Xóa phần tử khỏi mảng

1. Xóa phần tử khỏi mảng

Xóa phần tử khỏi mảng

```
int remove(int arr[], int n, int pos)
{
    if (n == 0)
        return 0;

    for (int i = pos; i < n-1; i++) {
        arr[i] = arr[i+1];
    }
    return n-1;
}
```

2. (Bài tập) Thêm đoạn mã đảm bảo giá trị **pos** hợp lệ trước khi xóa phần tử
3. (Bài tập) Viết hàm xóa **k** phần tử khỏi mảng tính từ vị trí **pos**.

5. Sắp xếp các phần tử trong mảng

1. Sắp xếp bằng

Sắp xếp các phần tử trong mảng

```
int sort(int a[], int n)
{
    for (int i = 0; i < n; i++) {
        for (int j = i+1; j < n; j++) {
            if (a[i] > a[j]) {
                int tmp = a[i];
                a[i] = a[j];
                a[j] = tmp;
            }
```

```
10            }
11        }
12    }
```

Hãy đọc kỹ, suy nghĩ về các câu lệnh trên:

- Dòng lệnh số 3, số 4 là 2 vòng for lồng nhau để duyệt cặp phần tử a[i], a[j] mà i < j (a[i] đứng trước a[j])
- Dòng lệnh số 5 kiểm tra xem thứ tự a[i] và a[j] có sai (a[i] > a[j])
- Nếu thứ tự bị sai, tráo đổi hai phần tử này bằng các dòng lệnh số 6, 7, 8

2. Nếu sử dụng **#include <algorithm>**, bạn có thể sắp xếp bằng chỉ một câu lệnh **sort** đơn giản của C++

Sắp xếp bằng lệnh sort của thư viện <algorithm>

```
sort(a, a+n);
```

3. (Bài tập) Viết hàm sắp xếp mảng các **string** theo thứ tự từ điển.
4. (Bài tập) Viết hàm sắp xếp mảng các số **double** theo thứ tự giảm dần.

6. Sử dụng mảng động <vector>

1. Mở hướng dẫn về **<vector>** trong **cplusplus.com**
2. Tìm hiểu cách sử dụng hàm **size()** của vector
3. Tìm hiểu cách sử dụng hàm **push_back()** của vector
4. Tìm hiểu cách sử dụng hàm **insert()** của vector
5. Tìm hiểu cách sử dụng hàm **pop_back()** của vector
6. Tìm hiểu cách sử dụng hàm **erase()** của vector

7. Thao tác với xâu

Khi sử dụng **string**, C++ cung cấp nhiều hàm cho phép thao tác với xâu ký tự như

- length(): độ dài xâu
- find(): tìm kiếm vị trí xâu con
- replace(): sửa một đoạn trong xâu
- substr(): lấy ra một xâu con

và còn nhiều hàm khác (mở hướng dẫn về **<string>** trong **cplusplus.com**)

Ví dụ thao tác với string

```
#include <iostream>
using namespace std;

int main()
{
    string hello = "Hello, Long";

    cout << hello << endl;
    cout << hello.length() << endl;
    for (int i = 0; i < hello.length(); i++) { // duyệt qua các ký tự
        cout << hello[i] << endl;
    }
```

```
    //hello.insert(hello.find("Long"), "Quoc ");
    string longName = "Long";
    int pos = hello.find(longName);  // tìm kiếm xâu con
    hello.replace(pos, longName.length(), "UET"); // thay thế xâu con
    cout << hello << endl;

    cout << hello.substr(1, 4) << endl;  // lấy xâu con

    cout << (longName > "Minh") << endl; // so sánh xâu theo thứ tự từ điển

    return 0;
}
```

1. (Bài tập) Đếm số từ trong xâu (các từ cách nhau bởi dấu cách hoặc dấu tab **\t** hoặc dấu xuống dòng **\n**.
2. (Bài tập) Chuẩn hóa tên: viết hoa đầu các từ
3. (Bài tập) Chuẩn hóa văn bản: viết hoa đầu các xâu, các khoảng trắng chỉ còn 1 dấu cách