ÔN TẬP CÁC THUẬT TOÁN SẮP XẾP

Bài 1: Thực hiện các yêu cầu sau

1. Viết chương trình tạo ra một mảng n phần tử là chuỗi ký tự ngẫu nhiên. Mỗi chuỗi ký tự có kích thước tối đa là 10 ký tự.

Cách 1: Sử dụng mảng `char` 2 chiều

```
#include <iostream>
#include <cstdlib>  // for rand() and srand()
#include <ctime>    // for time()
using namespace std;

const int MAX_LEN = 10; // maximum length of each string

int main() {
    int n;
    cout << "Enter the number of strings to generate: ";
    cin >> n;

    // Initialize the random number generator with the current time
    srand(time(0));

    // Define the characters that can be used in the strings
    const string chars =
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789";

    // Generate the array of strings
    // two-dimensional array to store the strings
    char arr[n][MAX_LEN+1];
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < MAX_LEN; j++) {
            // get a random index into the chars string
            int index = rand() % chars.length();
            // add the character at the random index to the string
            arr[i][j] = chars[index];
        }
        // add a null terminator to the end of the string
        arr[i][MAX_LEN] = '\0';
    }

    // Print the array of strings
    for (int i = 0; i < n; i++) {
        cout << arr[i] << endl;
    }

    return 0;
}
```

Cách 2: Sử dụng mảng String 1 chiều

```
#include <iostream>
#include <cstdlib>  // for rand() and srand()
#include <ctime>    // for time()
using namespace std;

const int MAX_LEN = 10; // maximum length of each string

int main() {
    int n;
    cout << "Enter the number of strings to generate: ";
    cin >> n;

    // Initialize the random number generator with the current time
    srand(time(0));

    // Define the characters that can be used in the strings
    const string chars =
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789";

    // Generate the array of strings
    string arr[n]; // create an array of n strings
    for (int i = 0; i < n; i++) {
        string s = "";
        for (int j = 0; j < 10; j++) {
            int index = rand() % chars.length(); // get a random index into the chars
string
            s += chars[index]; // add the character at the random index to the string
        }
        arr[i] = s; // set the ith element of the array to the generated string
    }

    // Print the array of strings
    for (int i = 0; i < n; i++) {
        cout << arr[i] << endl;
    }

    return 0;
}
```

2. Sắp xếp một mảng chuỗi kí tự theo thứ tự tăng dần sử dụng selection sort và merge sort.

Sử dụng selection sort

```
#include <iostream>
#include <cstdlib>
#include <ctime>
#include <string>

using namespace std;

const int MAX_LEN = 10;

// Function to generate a random string of length MAX_LEN
string generateRandomString() {
    // Define the characters that can be used in the string
    const string chars =
```

```
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789";

    // Generate the string one character at a time
    string s = "";
    for (int i = 0; i < MAX_LEN; i++) {
        int index = rand() % chars.length();
        s += chars[index];
    }
    return s;
}

// Function to print an array of strings
void printArray(string arr[], int n) {
    for (int i = 0; i < n; i++) {
        cout << arr[i] << endl;
    }
}

// Function to perform selection sort on an array of strings
void selectionSort(string arr[], int n) {
    int i, j, min_idx;

    for (i = 0; i < n-1; i++) {
        min_idx = i;
        for (j = i+1; j < n; j++) {
            if (arr[j] < arr[min_idx])
                min_idx = j;
        }

        swap(arr[min_idx], arr[i]);
    }
}

int main() {
    int n;
    cout << "Enter the number of strings to generate: ";
    cin >> n;

    // Initialize the random number generator
    srand(time(0));

    // Generate the array of strings
    string arr[n];
    for (int i = 0; i < n; i++) {
        arr[i] = generateRandomString();
    }

    // Print the unsorted array of strings
    cout << "Unsorted array:" << endl;
    printArray(arr, n);

    // Sort the array of strings using selection sort
    selectionSort(arr, n);

    // Print the sorted array of strings
    cout << "Sorted array:" << endl;
    printArray(arr, n);

    return 0;
}
```

Sử dụng merge sort

```
#include <iostream>
#include <cstdlib>
#include <ctime>
#include <string>

using namespace std;

const int MAX_LEN = 10;

// Function to generate a random string of length MAX_LEN
string generateRandomString() {
    // Define the characters that can be used in the string
    const string chars =
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789";

    // Generate the string one character at a time
    string s = "";
    for (int i = 0; i < MAX_LEN; i++) {
        int index = rand() % chars.length();
        s += chars[index];
    }
    return s;
}

// Function to print an array of strings
void printArray(string arr[], int n) {
    for (int i = 0; i < n; i++) {
        cout << arr[i] << endl;
    }
}

// Function to merge two sorted arrays into one sorted array
void merge(string arr[], string left[], int leftSize, string right[], int rightSize) {
    int i = 0, j = 0, k = 0;

    // Merge the two sorted arrays into one sorted array
    while (i < leftSize && j < rightSize) {
        if (left[i] < right[j]) {
            arr[k++] = left[i++];
        } else {
            arr[k++] = right[j++];
        }
    }

    // Copy any remaining elements from the left array to the sorted array
    while (i < leftSize) {
        arr[k++] = left[i++];
    }

    // Copy any remaining elements from the right array to the sorted array
    while (j < rightSize) {
        arr[k++] = right[j++];
    }
}


// Function to sort an array of strings using merge sort
void mergeSort(string arr[], int n) {
    if (n > 1) {
        int mid = n / 2;
```

```
        // Divide the array into two halves
        string left[mid];
        for (int i = 0; i < mid; i++) {
            left[i] = arr[i];
        }

        string right[n-mid];
        for (int i = mid; i < n; i++) {
            right[i-mid] = arr[i];
        }

        // Recursively sort each half of the array
        mergeSort(left, mid);
        mergeSort(right, n-mid);

        // Merge the two sorted halves into one sorted array
        merge(arr, left, mid, right, n-mid);
    }
}

int main() {
    int n;
    cout << "Enter the number of strings to generate: ";
    cin >> n;

    // Initialize the random number generator
    srand(time(0));

    // Generate the array of strings
    string arr[n];
    for (int i = 0; i < n; i++) {
        arr[i] = generateRandomString();
    }

    // Print the unsorted array of strings
    cout << "Unsorted array:" << endl;
    printArray(arr, n);

    // Sort the array of strings using merge sort
    mergeSort(arr, n);

    // Print the sorted array of strings
    cout << "Sorted array:" << endl;
    printArray(arr, n);

    return 0;
}
```

3. So sánh thời gian thực thi của hai thuật toán với chuỗi ký tự có kích thước là 10, 100 và 1000.

Bài 2: Viết chương trình thực hiện sắp xếp một chuỗi kí tự dựa trên độ dài của các chuỗi. Ví dụ, chuỗi ["dog", "cat", "elephant", "mouse"] sẽ được sắp xếp thành ["cat", "dog", "mouse", "elephant"].

Using vector

```cpp
#include <iostream>
#include <string>
#include <vector>

using namespace std;

// Function to sort an array of strings by length using insertion sort
void insertionSortByLength(vector<string>& arr) {
    for (int i = 1; i < arr.size(); i++) {
        string key = arr[i];
        int j = i - 1;

        // Move elements of arr[0..i-1], that are
        // greater than key, to one position ahead
        // of their current position
        while (j >= 0 && arr[j].length() > key.length()) {
            arr[j + 1] = arr[j];
            j = j - 1;
        }
        arr[j + 1] = key;
    }
}

int main() {
    vector<string> arr = {"dog", "cat", "elephant", "mouse"};

    // Print the unsorted array of strings
    cout << "Unsorted array:" << endl;
    for (int i = 0; i < arr.size(); i++) {
        cout << arr[i] << " ";
    }
    cout << endl;

    // Sort the array of strings by length using insertion sort
    insertionSortByLength(arr);

    // Print the sorted array of strings
    cout << "Sorted array:" << endl;
    for (int i = 0; i < arr.size(); i++) {
        cout << arr[i] << " ";
    }
    cout << endl;

    return 0;
}
```

Using array

```cpp
#include <iostream>
#include <string>

using namespace std;

// Function to sort an array of strings by length using insertion sort
void insertionSortByLength(string arr[], int n) {
    for (int i = 1; i < n; i++) {
        string key = arr[i];
        int j = i - 1;

        // Move elements of arr[0..i-1], that are
        // greater than key, to one position ahead
        // of their current position
        while (j >= 0 && arr[j].length() > key.length()) {
            arr[j + 1] = arr[j];
            j = j - 1;
        }
        arr[j + 1] = key;
    }
}

int main() {
    string arr[] = {"dog", "cat", "elephant", "mouse"};
    int n = sizeof(arr) / sizeof(arr[0]);

    // Print the unsorted array of strings
    cout << "Unsorted array:" << endl;
    for (int i = 0; i < n; i++) {
        cout << arr[i] << " ";
    }
    cout << endl;

    // Sort the array of strings by length using insertion sort
    insertionSortByLength(arr, n);

    // Print the sorted array of strings
    cout << "Sorted array:" << endl;
    for (int i = 0; i < n; i++) {
        cout << arr[i] << " ";
    }
    cout << endl;

    return 0;
}
```

Bài 3: Cho một mảng gồm n số nguyên phân biệt, hãy viết chương trình để tìm tất cả các cặp số nguyên có tổng bằng một giá trị sum được nhập vào từ bàn phím. So sánh thời gian thực thi của thuật toán trên với giải pháp vét cạn.

Bài 4: Đề xuất các thuật toán sắp xếp phù hợp cho các trường hợp sau và nêu lý do.

1. Dữ liệu nhỏ (n < 100).
2. Dữ liệu được sắp xếp theo chiều ngược lại với yêu cầu.

3. Dữ liệu gồm các phần tử ngẫu nhiên.
4. Dữ liệu có nhiều phần tử giống nhau.
5. Dữ liệu gồm các phần tử âm
6. Dữ liệu là mảng các chuỗi, mỗi chuỗi có độ dài là 10 kí tự .
7. Dữ liệu là mảng các chuỗi, mỗi chuỗi có độ dài là 1000 kí tự .

Bài 5: Một công ty vận chuyển cần sắp xếp một số lượng lớn các bưu kiện dựa trên mã bưu điện. Mỗi bưu kiện có thể nặng từ vài kí trở lên và việc xử lý thường xuyên có thể làm tăng nguy cơ hư hỏng. Đề xuất thuật toán sắp xếp giúp giảm thiểu số lần hoán đổi hoặc di chuyển bưu kiện xuống mức thấp nhất.