Họ và tên: Hoàng Tấn Lợi

Lớp: COMP140104

Bài 4: Cài đặt thuật toán Quicksort

* Đầu vào: Mảng các phần tử chưa sắp xếp
* Đầu ra: Mảng các phần tử thoả điều kiện arr[i] <= arr[j] với mọi i < j
* Cách xử lí: Chọn phần tử ở giữa làm phần tử chốt (pivot), ta chuyển các phần tử nhỏ hơn pivot về phía bên trái, lớn hơn pivot về phía bên phải. Sau khi thực hiện xong thì phần tử pivot đã ở đúng vị trí. Ta thực hiện đệ quy trên hai dãy con bên trái và bên phải pivot cho đến khi dãy chỉ còn 1 phần tử. Độ phức tạp trung bình của thuật toán là O(nlogn). Độ phức tạp thuật toán trong trường hợp xấu nhất là O(n^2).

Source code:

void quicksort(int\* arr, int begin, int end)

{

if (begin < end) //nếu như vị trí bắt đầu lớn hơn kết thúc thì không cần thực hiện

{

int pivot = arr[(high + low)/2]; //phần tử chốt nằm giữa dãy

int i = low - 1;

int j = high + 1;

while (true) //vòng lặp

{

while (arr[++i] < pivot); //khi phần tử bên trái nhỏ hơn pivot thì tiếp tục

while (arr[--j] > pivot); //khi phần tử bên phải lớn hơn pivot thì tiếp tục

if (i >= j) break; //nếu như i>=j, tức là đã đi qua pivot thì thôi

std::swap(arr[i], arr[j]); //hoán đổi vị trí của phần tử i và j

}

quicksort(arr, begin, j); //thực hiện đệ quy trên dãy bên trái pivot

quicksort(arr, j + 1, end); //thực hiện đệ quy trên dãy bên phải pivot

}

}

Bài 5: Cài đặt thuật toán Quicksort với phần tử chốt ngẫu nhiên

* Đầu vào: Mảng các phần tử chưa sắp xếp
* Đầu ra: Mảng các phần tử thoả điều kiện arr[i] <= arr[j] với mọi i < j
* Cách xử lí: Tương tự như thuật toán Quicksort, tuy nhiên phần chọn phần tử chốt sẽ random từ đầu đến cuối dãy, tránh rơi vào trường hợp xấu nhất có độ phức tạp O(n^2)

Source code:

#include <cstdlib> //for rand(), srand()

#include <ctime> //for time()

void quickSort(int \*arr, int begin, int end)

{

if (begin < end) //nếu như vị trí bắt đầu lớn hơn kết thúc thì không cần thực hiện

{

int pivot = arr[begin + std::rand() % (end - begin + 1)]; //phần tử chốt nằm ngẫu nhiên trong dãy

int i = begin - 1;

int j = end + 1;

while (true)

{

while (arr[++i] < pivot); //khi phần tử bên trái nhỏ hơn pivot thì tiếp tục

while (arr[--j] > pivot); //khi phần tử bên phải lớn hơn pivot thì tiếp tục

if (i >= j) break; //nếu như i>=j, tức là đã đi qua pivot thì thôi

std::swap(arr[i], arr[j]); //hoán đổi vị trí của phần tử i và j

}

quickSort(arr, low, j); //thực hiện đệ quy trên dãy bên trái pivot

quickSort(arr, j + 1, high); //thực hiện đệ quy trên dãy bên phải pivot

}

}

Bài 6: Cài đặt thuật toán Tháp Hà Nội:

Nhiệm vụ của trò chơi là di chuyển các đĩa có kích cỡ khác nhau sang cột khác sao cho vẫn đảm bảo thứ tự ban đầu của các đĩa: đĩa nhỏ nằm trên đĩa lớn. Dưới đây là một số qui tắc cho trò chơi toán học Tháp Hà Nội (Tower of Hanoi):

* Mỗi lần chỉ có thể di chuyển một đĩa từ cột này sang cột khác.
* Chỉ được di chuyển đĩa nằm trên cùng (không được di chuyển các đĩa nằm giữa).
* Đĩa có kích thước lớn hơn không thể được đặt trên đĩa có kích thước nhỏ hơn.

Giải quyết bài toán:

* Đầu vào: n đĩa theo thứ tự từ nhỏ đến lớn, sắp xếp ở cột A
* Đầu ra: n đĩa theo thứ tự từ nhỏ đến lớn, ở cột C
* Cách giải quyết bài toán: Bài toán được thực hiện đệ quy như sau:
  + Trường hợp cơ bản: Nếu chỉ có một đĩa, ta chuyển nó từ cột A sang cột C ngay lập tức.
  + Di chuyển n-1 đĩa từ cột A sang cột B.
  + Di chuyển đĩa thứ n từ cột A sang cột C.
  + Di chuyển n-1 đĩa còn lại từ cột B sang cột C.

Source code:

#include <iostream>

void moveDisk(int disk, char source, char destination)

{

std::cout << "Move disk " << disk << " from " << source << " to " << destination << std::endl; //di chuyển đĩa thứ disk từ cột nguồn sang cột đích

}

void HanoiTower(int disks, char source, char auxilary, char destination)

{

if (disks == 1) //trường hợp cơ bản: chỉ có 1 đĩa

{

moveDisk(disks, source, destination);

return;

}

HanoiTower(disks - 1, source, destination, auxilary); //chuyển n-1 đĩa từ cột nguồn sang cột tạm

moveDisk(disks, source, destination); //chuyển đĩa thứ n từ cột nguồn sang cột đích

HanoiTower(disks - 1, auxilary, source, destination); //chuyển n-1 đĩa từ cột tạm sang cột đích

}

Bài 7: Cài đặt bài toán 8 quân hậu:

Bài toán tám quân hậu là bài toán đặt tám quân hậu trên bàn cờ vua kích thước 8×8 sao cho không có hai quân nào đứng trên cùng hàng, hoặc cùng cột hoặc cùng đường chéo. Bài toán tám quân hậu có thể tổng quát hóa thành bài toán đặt n quân hậu trên bàn cờ n×n(n ≥ 4).

Giải quyết bài toán:

* Đầu vào: Bàn cờ 8x8 trống, chưa có quân nào
* Đầu ra: Một cách để sắp xếp 8 quân hậu theo điều kiện bài toán
* Cách giải quyết bài toán: Sử dụng đệ quy quay lui: Kiểm tra nếu như vị trí của quân hậu hiện tại không có quân nào ăn, ta đặt một quân hậu khác vào vị trí hàng tiếp theo, nếu không thì ta quay lại vị trí trước.

Source code:

bool validate(int row, int col) {

for (int i = 0; i < row; i++) //check các hàng xem đã có quân hậu nào chưa

if (board[i][col] == 1) return 0;

for (int i = row, j = col; i >= 0 && j >= 0; i--, j--) //check đường chéo từ góc trái trên sang phải dưới

if (board[i][j] == 1) return 0;

for (int i = row, j = col; i >= 0 && j < boardSize; i--, j++) //check đường chéo từ góc trái dưới sang phải trên

if (board[i][j] == 1) return 0;

return 1; //nếu check xong mà không có quân, tức là có thể đặt một quân hậu vào vị trí đó

}

bool Try(int row) {

if (row >= boardSize) return 1;

for (int col = 0; col < boardSize; ++col)

{

if (validate(row, col)) //kiểm tra vị trí hiện tại có đặt được không

{

board[row][col] = 1; //đặt 1 con hậu vào vị trí đó

if (Try(row + 1)) return 1; //thử đặt tiếp 1 con hậu vào vị trí row+1

board[row][col] = 0; // quay lui

}

}

return 0;

}