|  |  |
| --- | --- |
|  | **2017** |
|  | Trường Đại Học Khoa Học Tự Nhiên  Khoa Công Nghệ Thông Tin Môn: Kiến trúc máy tính và hợp ngữGiáo viên hướng dẫn: Nguyễn Viết Long |

|  |
| --- |
| ***Report project II MIPS architecture and assembly language*** |
| [Báo cáo quá trình triển khai hai thuật toán sắp xếp Selection Sort và Insertion Sort bằng MIPS 32 bit ] By: Phùng Thị Hòa (1512186)  Bùi Thị Hồng (1512188)  Hà Phước Huy (1512195) |
|  |
|  |

Mục Lục

[I. Phân công công việc 2](#_Toc481178024)

[II. Môi trường lập trình 2](#_Toc481178025)

[III. Ý tưởng thiết kế 2](#_Toc481178026)

[IV. Quá trình thực hiện 3](#_Toc481178027)

[V. Phạm vi biểu diễn của dữ liệu 4](#_Toc481178028)

[VI. Kiểm thử 4](#_Toc481178029)

[VII. Mức độ hoàn thành 5](#_Toc481178030)

[VIII. Nguồn tham khảo 5](#_Toc481178031)

# Phân công công việc

Bảng thông tin cá nhân và phân công công việc:

|  |  |  |
| --- | --- | --- |
| Họ và tên | MSSV | Công việc |
| Phùng Thị Hòa | 1512186 | Viết hàm sắp xếp Selection Sort và Insertion Sort.  Kiếm tra và test chương trình. |
| Bùi Thị Hồng | 1512188 | Viết báo cáo. |
| Hà Phước Huy | 1512195 | Viết hàm đọc, xuất file.  Tổng hợp các phần, tạo chương trình hoàn chỉnh. |

# Môi trường lập trình

* Ngôn ngữ lập trình: MIPS 32 bit
* Môi trường lập trình: MARS

# Ý tưởng thiết kế

* Selection sort trong những thuật toán sắp xếp đơn giản nhất. Ý tưởng cơ bản của cách sắp xếp này là:
* Lượt đầu tiên, ta chọn phần tử nhất trong mảng từ vị trí 0 đến cuối mảng và đổi giá trị của phần tử nhỏ nhất với phần tử ở vị trí đầu tiên (vị trí 0).
* Ở lượt thứ hai, ta chọn phần tử nhỏ nhất ở vị trí 1 đến cuối mảng, đổi giá trị phần tử nhỏ nhất với phần tử ở vị trí số 1.
* ...
* Ta tiến hành lặp lại thao tác cho đến hết mảng.
* Xét mảng a[0...n]. Ta thấy dãy con chỉ gồm một phần tử a[0] có thể coi là một dãy đã sắp xếp rồi. Xét thêm a[1], ta so sánh giữa a[0] và a[1], nếu thấy a[1]<a[0] thì ta chèn nó trước a[0]. Đối với a[2], ta chỉ dãy gồm a[0],a[1] đã được sắp xếp, ta tìm cách chèn a[2] sao cho phù hợp nhất để được thứ tự sắp xếp. Nói một cách tổng quát ta xếp mảng a[0...i] trong điều kiện mảng a[0...i-1] đã được sắp xếp rồi, bằng cách chèn a[i] vào chị trí đúng, để được toàn bộ mảng được sắp xếp phù hợp.
* Các thao tác với file, trong MIPS dữ liệu được đọc từ file được lưu dưới dạng chuỗi vì vậy muốn sắp xếp được cần phải đọc từng ký tự, rồi chuyển các ký tự này số nguyên lưu vào mảng, lúc này mới thực hiện tính toán được. Khi ghi kết quả ra file, cũng cần chuyển các số nguyên ở mảng đã được sắp xếp thành ký tự rồi mới lưu sang file.

# Quá trình thực hiện

* Đầu tiên, viết hai thuật toán sắp xếp là Selection Sort và Insertion Sort:
* Tìm hiểu ý tưởng của hai thuật toán.
* Thực hiện ý tưởng đó bằng C++
* Dựa trên các file demo và các nguồn tham khảo để thực hiện chuyển từ ngôn ngữ C++ sang MIPS.
* Kết hợp hai hàm sắp xếp viết được với file Mang.asm của thầy để tiến hành nhập dữ liệu, nhập dữ liệu và quan sát kết quả, kiểm tra tính đúng sai, cách vận hành của hai thuật toán sắp xếp được yêu cầu.
* Viết hàm đọc, xuất file:

1. Mở, đọc, ghi, đóng file với syscall

|  |  |  |  |
| --- | --- | --- | --- |
| Service | Mã trong $v0 | Tham số | Kết quả |
| Mở (open) file | 13 | $a0 = địa chỉ chuỗi chứa tên file  $a1 = flags (0: đọc, 1: ghi) | $v0 chứa file decriptor |
| Đọc (read) file | 14 | $a0 = file descriptor  $a1 = địa chỉ của buffer  $a2 = số kí tự lớn nhất đễ đọc | $v0 chứa số kí tự đã đọc |
| Ghi (write) file | 15 | $a0 = file descriptor  $a1 = địa chỉ của buffer  $a2 = số kí tự cần ghi | $v0 chứa số kí tự đã ghi |
| Đóng (close) file | 16 | $a0 = file descriptor |  |

-Với MIPS, việc đọc và ghi file chỉ hỗ trợ đọc và ghi kí tự(chuỗi). Vì thế, để đọc được số nguyên từ file, ta cần tách từng chuỗi số nguyên từ chuỗi đã đọc ra từ file và chuyển chúng thành số nguyên. Ngược lại, ta cũng cần chuyển các số nguyên thành chuỗi kí tự và in ra file.

⇨Cần viết lại hàm **stoi** giúp chuyển chuỗi số nguyên thành số nguyên để lưu vào mảng arr. Song song đó, ta cũng cần viết lại hàm **to\_string** giúp chuyển các số nguyên từ mảng arr thành chuỗi các số nguyên và ghi vào buffer sau đó dùng buffer ghi vào file output.

1. Hàm stoi (chuyển toàn bộ chuỗi thành các phần tử int của mảng)

int j = 0;

for (int i = 0; i < n; i++){

int tmp = 0;

while (str[j] >= '0' && str[j] <= '9')

{

temp \*= 10; //$temp = temp << 1 + temp << 3

temp += str[j++] - '0';

}

j++; //Bỏ qua dấu ‘ ‘

arr[i] = temp;

}

1. Hàm to\_string (chuyển toàn mảng thành buffer):

for (int i = 0; i < n; i++){

int x = array[i];

Stack.push(-1); //Đặt lính canh

do

{

t = x % 10;

x = x / 10;

stack.push(t); //Bỏ vào $sp

} while (x == 0);

while (stack[i] != -1){ //Gặp lính canh thì dừng lại

temp = stack.top();

stack.pop();

temp = temp + '0';

str += temp;

}

Stack.pop(); //Bỏ qua lính canh

str += ' ';

}

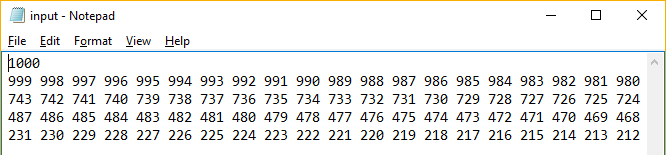
* Tiến hành tổng kết hợp giữa các hàm đọc ghi file với các thuật toán sắp xếp.
* Kiểm tra tính đúng đắn của chương trình.
* Viết báo cáo tống kết.

# Phạm vi biểu diễn của dữ liệu

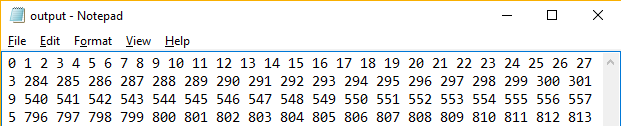
* Có thể biễu diễn số nguyên tối đa 10 chữ số cho 1 số int và 1000 phần tử của mảng.

# Kiểm thử

* Input.txt:



* Output.txt:



# Mức độ hoàn thành

Hoàn thành 95% yêu cầu của đồ án.

* Đọc xuất được file.
* Các thuật toán Sort chạy ra đúng kết quả mong muốn.

# Nguồn tham khảo

* http://chortle.ccsu.edu/AssemblyTutorial/index.html
* https://www.scribd.com/document/3577342/MIPS-assembly-language-programming
* http://dkrizanc.web.wesleyan.edu/courses/231/07/mips-spim.pdf
* http://logos.cs.uic.edu/366/notes/MIPS%20Quick%20Tutorial.htm
* http://www.cs.cornell.edu/~tomf/notes/cps104/mips.html
* Các file demo của thầy, slide của cô.