

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH



KIẾN TRÚC MÁY TÍNH (CO2007)

Báo cáo Bài tập lớn

SELECTION SORT SỐ NGUYÊN

GVHD: ThS. Nguyễn Xuân Minh

Lớp: L07

Nhóm: 122

Sinh viên: **2311572 – Đặng Ngọc Khiêu**

2311525 – Nguyễn Vũ Duy Khánh

Thành phố Hồ Chí Minh, 17 tháng 11 năm 2024



Mục lục

1	Đề bài	2
2	Selection sort số nguyên	2
2.1	Selection sort	2
2.2	Giải pháp hiện thực	2
2.3	Giải thuật	3
2.4	Đánh giá chương trình	4
2.4.1	Tổng kê số lệnh sử dụng trong chương trình	4
2.4.2	Thời gian thực thi của chương trình	5
2.4.3	Kiểm thử chương trình	6



1 Đề bài

$$(122 - 1) \% 9 + 1 = 5$$

Đề 5: Viết chương trình sắp xếp dãy số nguyên có 10 phần tử dùng giải thuật Selection sort. Yêu cầu xuất ra màn hình các bước có thay đổi thứ tự trong dãy. Dữ liệu đầu vào đọc từ file lưu trữ dạng nhị phân trên đĩa INT10.BIN (10 phần tử x 4 bytes = 40 bytes).

2 Selection sort số nguyên

2.1 Selection sort

Selection sort (*sắp xếp chọn*) là giải thuật sắp xếp dựa trên việc so sánh in-place, trong đó danh sách được chia thành hai phần, phần được sắp xếp (*sorted list*) ở bên trái và phần chưa được sắp xếp (*unsorted list*) ở bên phải. Ban đầu, phần được sắp xếp là trống và phần chưa được sắp xếp là toàn bộ danh sách ban đầu. Phần tử nhỏ nhất được lựa chọn từ mảng chưa được sắp xếp và được trao đổi với phần bên trái nhất và phần tử đó trở thành phần tử của mảng được sắp xếp. Tiến trình này tiếp tục cho tới khi toàn bộ từng phần tử trong mảng chưa được sắp xếp đều được di chuyển sang mảng đã được sắp xếp.

Các bước thực hiện giải thuật selection sort:

1. Tìm phần tử nhỏ nhất trong mảng và đổi chỗ nó với phần tử ở vị trí đầu tiên;
2. Tìm phần tử nhỏ nhất trong mảng con từ vị trí thứ hai trở đi và đổi chỗ nó với phần tử ở vị trí thứ hai;
3. Tiếp tục quá trình trên cho đến khi toàn bộ mảng được sắp xếp.

2.2 Giải pháp hiện thực

Chương trình được viết và chạy trên MARS MIPS 4.5.

Trước tiên, ta viết chương trình tạo file dữ liệu (chương trình này không thuộc file mã nguồn, đoạn chương trình tạo file nằm ở phần cuối bài báo cáo này):

- Tạo một mảng kiểu số nguyên với 10 phần tử bất kỳ;



- Tạo một chuỗi có tên “INT10.BIN” để làm tên của file dữ liệu;
- Sử dụng chức năng syscall số 13 để mở file với cờ (flag) = 1 để ghi dữ liệu;
- Load địa chỉ của mảng vào thanh ghi \$a0 (đóng vai trò như bộ đệm);
- Sử dụng chức năng syscall số 15 để ghi dữ liệu từ bộ đệm vào file;
- Sử dụng chức năng syscall số 16 để đóng file.

Sau khi đã có file dữ liệu có tên INT10.BIN, ta viết chương trình đọc file vừa tạo và thực hiện sắp xếp chọn trên các phần tử có trong file.

Các bước đọc file:

- Tạo một bộ đệm có kích thước 40 bytes;
- Tạo một chuỗi có tên “INT10.BIN” là tên của file dữ liệu;
- Sử dụng chức năng syscall số 13 để mở file với cờ (flag) = 0 để đọc dữ liệu;
- Load địa chỉ của bộ đệm vào thanh ghi \$a0;
- Sử dụng chức năng syscall số 14 để đọc dữ liệu từ file vào bộ đệm;
- Sử dụng chức năng syscall số 16 để đóng file.

Thực hiện sắp xếp chọn đối với mảng đọc được từ file (bài này nhóm chúng em lựa chọn sắp xếp giá trị các phần tử tăng dần).

2.3 Giải thuật

Giải thuật được mô tả bằng mã giả sau:

```
1 current = 0
2 while current < count - 1 do
3     smallest = current
4     walker = current + 1
5     while walker < count do
6         if data[walker] < data[smallest] then
7             smallest = walker
8         end
9         walker = walker + 1
```



```
10  end
11  swap(current, smalest)
12  current = current + 1
13 end
```

Dựa trên mã giả, nhóm có thể viết được chương trình trên MARS MIPS 4.5 gián tiếp qua chương trình C++ dưới đây.

```
1 void selectionSort(int* arr, int n) {
2     for(int i = 0; i < n - 1; i++) {
3         int indexOfMin = i;
4         for(int j = i + 1; j < n; j++) {
5             if(arr[j] < arr[indexOfMin]) indexOfMin = j;
6         }
7         if(indexOfMin != i) {
8             int tmp = arr[indexOfMin];
9             arr[indexOfMin] = arr[i];
10            arr[i] = tmp;
11            printArray(arr, n);
12            cout << '\n';
13        }
14    }
15 }
```

2.4 Đánh giá chương trình

2.4.1 Thống kê số lệnh sử dụng trong chương trình

Trong chương trình mô phỏng hợp ngữ assembly MIPS của nhóm, đã sử dụng tổng cộng các lệnh R, I, J là

Loại lệnh	Số lệnh
R	26
I	65
J	9
Tổng số	100



2.4.2 Thời gian thực thi của chương trình

Clock Rate = 1 GHz.

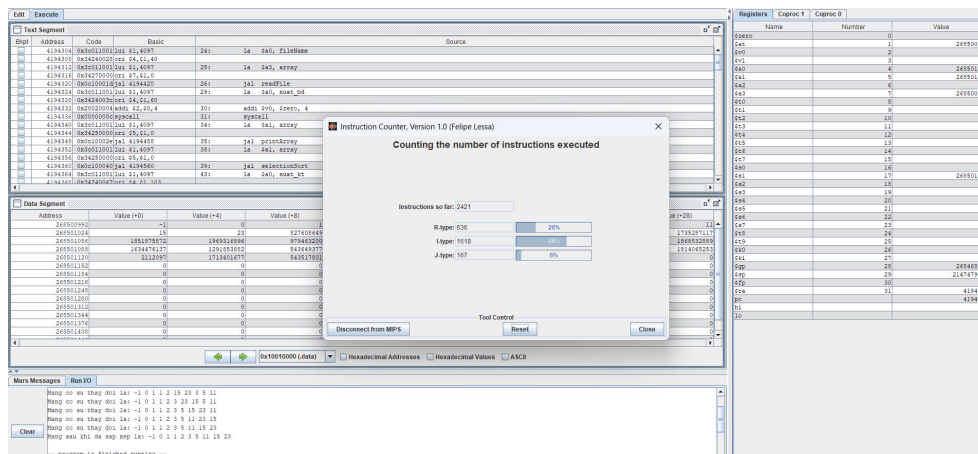
Theo kiến trúc Pipeline, ta có công thức tìm số chu kỳ thực thi

$$\text{số chu kỳ thực thi} = \text{số lệnh} + \text{số giai đoạn} - 1$$

CPI được tính theo công thức

$$\text{CPI} = \frac{\text{số chu kỳ thực thi}}{\text{số lệnh}} = \frac{\text{số lệnh} + \text{số giai đoạn} - 1}{\text{số lệnh}}$$

Theo ảnh dưới, sau khi nhóm sử dụng công cụ xác định số lệnh thực thi của chương trình, nhóm thu được số lệnh là 2421.

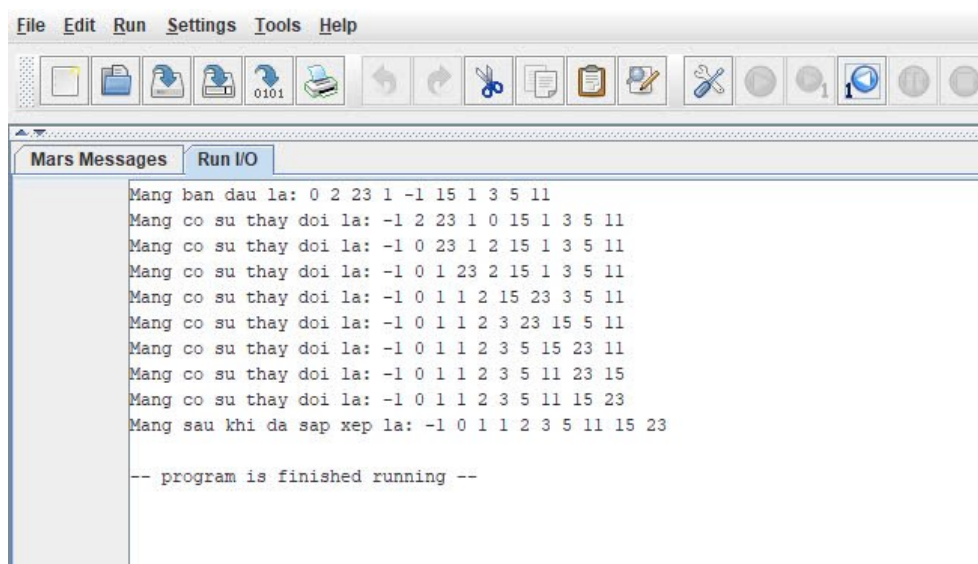


Giả định số giai đoạn của một lệnh theo kiến trúc Pipeline là 5, thời gian chạy của chương trình được tính theo công thức.

$$\begin{aligned} \text{CPU execution time} &= \frac{\text{số lệnh} \cdot \text{CPI}}{\text{Clock Rate}} = \frac{\text{số lệnh} \cdot \frac{\text{số lệnh} + \text{số giai đoạn} - 1}{\text{số lệnh}}}{\text{Clock Rate}} \\ &= \frac{2421 + (5 - 1)}{1 \cdot 10^9} \\ &= 2.425 \cdot 10^{-6} \text{ s} \end{aligned}$$

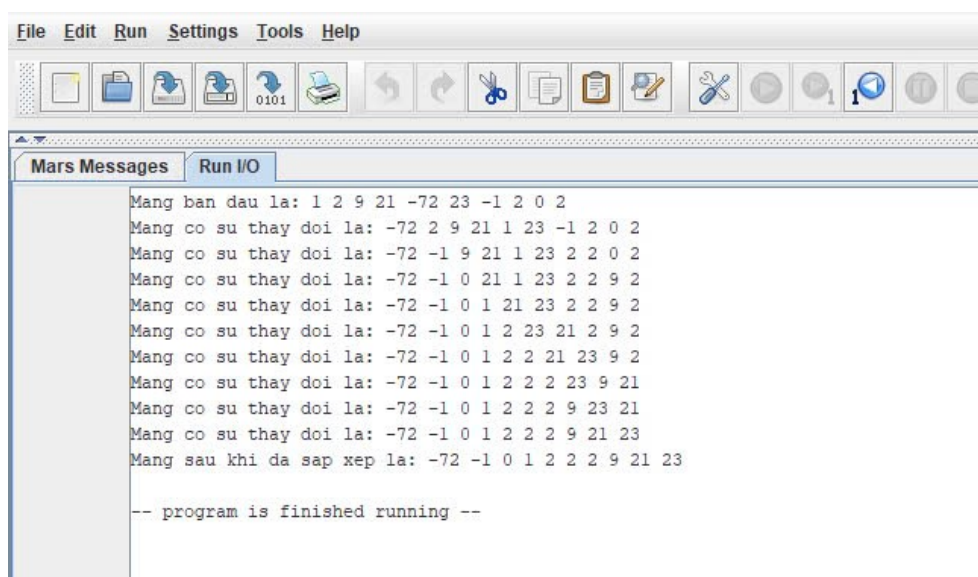
2.4.3 Kiểm thử chương trình

Ba kết quả sau khi kiểm thử chương trình:



```
File Edit Run Settings Tools Help
Mars Messages Run I/O
Mang ban dau la: 0 2 23 1 -1 15 1 3 5 11
Mang co su thay doi la: -1 2 23 1 0 15 1 3 5 11
Mang co su thay doi la: -1 0 23 1 2 15 1 3 5 11
Mang co su thay doi la: -1 0 1 23 2 15 1 3 5 11
Mang co su thay doi la: -1 0 1 1 2 15 23 3 5 11
Mang co su thay doi la: -1 0 1 1 2 3 23 15 5 11
Mang co su thay doi la: -1 0 1 1 2 3 5 15 23 11
Mang co su thay doi la: -1 0 1 1 2 3 5 11 23 15
Mang co su thay doi la: -1 0 1 1 2 3 5 11 15 23
Mang sau khi da sap xep la: -1 0 1 1 2 3 5 11 15 23
-- program is finished running --
```

Figure 1: Test case 1: 0, 2, 23, 1, -1, 15, 1, 3, 5, 11



```
File Edit Run Settings Tools Help
Mars Messages Run I/O
Mang ban dau la: 1 2 9 21 -72 23 -1 2 0 2
Mang co su thay doi la: -72 2 9 21 1 23 -1 2 0 2
Mang co su thay doi la: -72 -1 9 21 1 23 2 2 0 2
Mang co su thay doi la: -72 -1 0 21 1 23 2 2 9 2
Mang co su thay doi la: -72 -1 0 1 21 23 2 2 9 2
Mang co su thay doi la: -72 -1 0 1 2 23 21 2 9 2
Mang co su thay doi la: -72 -1 0 1 2 2 21 23 9 2
Mang co su thay doi la: -72 -1 0 1 2 2 2 23 9 21
Mang co su thay doi la: -72 -1 0 1 2 2 2 9 23 21
Mang co su thay doi la: -72 -1 0 1 2 2 2 9 21 23
Mang sau khi da sap xep la: -72 -1 0 1 2 2 2 9 21 23
-- program is finished running --
```

Figure 2: Test case 2: 1, 2, 9, 21, -72, 23, -1, 2, 0, 2

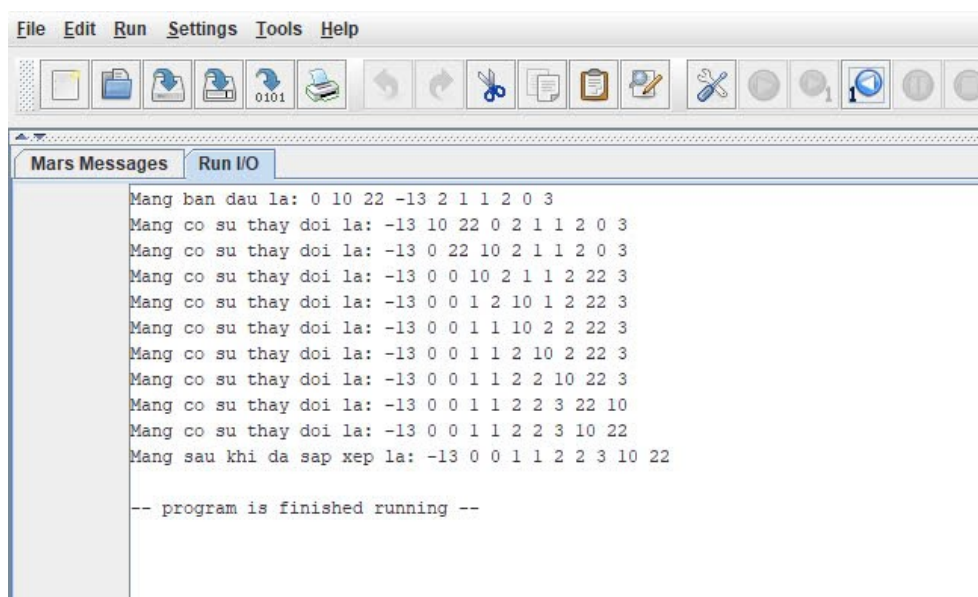


Figure 3: Test case 3: 0, 10, 22, -13, 2, 1, 1, 2, 0, 3

Dưới đây là đoạn chương trình tạo file dữ liệu:

```

1 #Chương trình: Tao file
2 .data
3 #Cac dinh nghia bien
4 array: .word 0 2 23 1 -1 15 1 3 5 11
5
6 flag: .word 0
7 Ten_File: .ascii "INT10.BIN"
8 #Cac cau nhac nhap du lieu
9 str_tc: .ascii "Thanh cong."
10 str_loi: .ascii "Mo file bi loi."
11
12 #Code segment
13 .text
14 .globl main
15 main:
16 #Nhap (syscall)
17 move $t0, $a0
18 #Xu ly

```




```
19 #mo file
20  la $a0, Ten_File
21  addi $a1, $zero, 1 # open with write-only
22  li $v0, 13
23  syscall
24  bltz $v0, bao_loi
25  sw $v0, flag
26
27 # ghi vao file
28  la  $a1, array
29
30  lw  $a0, flag #file descriptor
31  li  $a2, 40
32  li  $v0, 15
33  syscall
34
35 #dong file
36  li $v0, 16
37  syscall
38
39  la $a0, str_tc
40  li $v0, 4
41  syscall
42
43  j  Ket_thuc
44 #Xuat ket qua (syscall)
45 bao_loi:
46  la $a0, str_loi
47  li $v0, 4
48  syscall
49 #ket thuc chương trình (syscall)
50 Ket_thuc:
51  addiu $v0, $zero, 10
52  syscall
```