**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KỸ THUẬT MÁY TÍNH**

**TỔ CHỨC VÀ CẤU TRÚC MÁY TÍNH II (IT012)**

**A blue logo with a planet and text

Description automatically generated**

**Sinh viên: Trần Nguyễn Thái Bình**

**MSSV: 23520161**

**Giảng viên hướng dẫn: Nguyễn Thành Nhân**

**IT012.O21.1 – Lab 05**

**MỤC LỤC**

[1. Thao tác với mảng 3](#_Toc166878361)

[2. Thao tác với con trỏ 3](#_Toc166878362)

[3. Bài tập *(chỉ sử dụng con trỏ)* 3](#_Toc166878363)

[3.1. Nhập một mảng các số nguyên n phần tử *(nhập vào số phần tử và giá trị của từng phần tử)*, xuất ra cửa sổ I/O của MARS theo từng yêu cầu sau: 3](#_Toc166878364)

[3.2. Nhập một mảng các số nguyên n phần tử *(nhập vào số phần tử và giá trị của từng phần tử)*. Mảng này gọi là A. 3](#_Toc166878365)

1. **Thao tác với mảng**

* Thao tác với mảng Mảng với n phần tử là một chuỗi n phần tử liên tiếp nhau trong bộ nhớ. Thao tác với mảng trong MIPS là thao tác trực tiếp với byte/word trong bộ nhớ.
* Để cấp phát chuỗi word hoặc byte trong bộ nhớ, có giá trị khởi tao sử dụng “.word” hoặc “.byte” trong “.data”
* Để cấp phát chuỗi byte không có giá trị khởi tạo trước, sử dụng “.space” trong “.data”
* Cho ba mảng với cấp phát dữ liệu trong bộ nhớ như sau:

1. .data

2. array1: .word 5, 6, 7, 8, 1, 2, 3, 9, 10, 4

3. size1: .word 10

4.

5. array2: .byte 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16

6. size2: .word 16

7.

8. array3: .space 8

9. size3: .word 8

* Mảng array1 có 10 word, kích thước được lưu trong size1; Mảng array2 có 16 byte, kích thước được lưu trong size2; Mảng array3 có 8 byte, kích thước được lưu trong size3.

Viết code trong phần “.text” thực hiện riêng từng phần việc:

* In ra cửa sổ I/O của MARS tất cả các phần tử của mảng array1 và array2
* Gán các giá trị cho mảng array3 sao cho

array3[i] = array2[i] + array2[size2 - 1 - i]

* + Người sử dụng nhập vào mảng thứ mấy và chỉ số phần tử cần lấy trong mảng đó, chương trình xuất ra phần tử tương ứng.
* Chương trình ASM:

1. .data

2. array1: .word 5, 6, 7, 8, 1, 2, 3, 9, 10, 4

3. size1: .word 10

4. array2: .byte 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16

5. size2: .word 16

6. array3: .space 8

7. size3: .word 8

8. str\_print\_arr\_1: .asciiz "xuat mang 1:"

9. str\_print\_arr\_2: .asciiz "xuat mang 2:"

10. str\_print\_arr\_3: .asciiz "xuat mang 3:"

11. str\_print\_prompt\_input: .asciiz "Nhap thu tu mang can truy xuat: "

12. str\_print\_prompt\_input\_index: .asciiz "Nhap index cua mang: "

13. tab\_char: .asciiz "\t"

14. end\_line\_char: .asciiz "\n"

15.

16. .text

17.

18. main:

19.

20. print\_arr\_1:

21. la $a0, str\_print\_arr\_1

22. li $v0, 4

23. syscall

24. jal print\_end\_line

25. la $s1, array1

26. lw $s2, size1

27.

28. li $t0, 0

29.

30. loop\_print\_arr\_1:

31. bge $t0, $s2, continue\_1

32. sll $t1, $t0, 2

33. add $t1, $t1, $s1

34. lw $a0, 0($t1)

35. li $v0, 1

36. syscall

37. addi $t0, $t0, 1

38. jal print\_tab

39. j loop\_print\_arr\_1

40.

41. continue\_1:

42. jal print\_end\_line

43.

44. print\_arr\_2:

45. la $a0, str\_print\_arr\_2

46. li $v0, 4

47. syscall

48. jal print\_end\_line

49. la $s1, array2

50. lw $s2, size2

51.

52. li $t0, 0

53.

54. loop\_print\_arr\_2:

55. bge $t0, $s2, continue\_2

56. add $t1, $t0, $s1

57. lb $a0, 0($t1)

58. li $v0, 1

59. syscall

60. addi $t0, $t0, 1

61. jal print\_tab

62. j loop\_print\_arr\_2

63.

64. continue\_2:

65. jal print\_end\_line

66.

67. prepare\_arr\_3:

68. la $s1, array2

69. lw $s2, size2

70. lw **$s3**, size2

71. add **$s3**, **$s3**, $s1

72. subi **$s3**, **$s3**, 1

73. la **$s4**, array3

74. lw **$s5**, size3

75. move $s6, **$s4**

76. li $t0, 0

77. move $t1, **$s3**

78.

79. loop\_process\_arr\_3:

80. bge $t0, **$s5**, continue\_3

81. add $t2, $s1, $t0

82. lb $t3, 0($t2)

83. lb $t4, 0($t1)

84. add $t5, $t3, $t4

85. sb $t5, 0($s6)

86. addi $t0, $t0, 1

87. subi $t1, $t1, 1

88. addi $s6, $s6, 1

89. j loop\_process\_arr\_3

90.

91. continue\_3:

92. jal print\_end\_line

93.

94. prepare\_print\_arr\_3:

95. la $a0, str\_print\_arr\_3

96. li $v0, 4

97. syscall

98. jal print\_end\_line

99. la $s1, array3

100. lw $s2, size3

101.

102. li $t0, 0

103.

104. loop\_print\_arr\_3:

105. bge $t0, $s2, continue\_4

106. add $t1, $t0, $s1

107. lb $a0, 0($t1)

108. li $v0, 1

109. syscall

110. addi $t0, $t0, 1

111. jal print\_tab

112. j loop\_print\_arr\_3

113.

114. continue\_4:

115. jal print\_end\_line

116.

117. prompt\_user\_input\_which\_arr:

118. la $a0, str\_print\_prompt\_input

119. li $v0, 4

120. syscall

121. li $v0, 5

122. syscall

123. move $s0, $v0

124.

125. prompt\_user\_input\_which\_index:

126. la $a0, str\_print\_prompt\_input\_index

127. li $v0, 4

128. syscall

129. li $v0, 5

130. syscall

131. move $s1, $v0

132.

133. compare\_input:

134. li $t0, 1

135. beq $s0, $t0, get\_ele\_of\_arr\_1

136. li $t0, 2

137. beq $s0, $t0, get\_ele\_of\_arr\_2

138. li $t0, 3

139. beq $s0, $t0, get\_ele\_of\_arr\_3

140.

141. get\_ele\_of\_arr\_1:

142. la **$s3**, array1

143. sll $s1, $s1, 2

144. add **$s4**, **$s3**, $s1

145. lw **$s5**, 0(**$s4**)

146. j print\_got\_ele

147.

148. get\_ele\_of\_arr\_2:

149. la **$s3**, array2

150. add **$s4**, **$s3**, $s1

151. lb **$s5**, 0(**$s4**)

152. j print\_got\_ele

153.

154. get\_ele\_of\_arr\_3:

155. la **$s3**, array3

156. add **$s4**, **$s3**, $s1

157. lb **$s5**, 0(**$s4**)

158. j print\_got\_ele

159.

160. print\_got\_ele:

161.

162. move $a0, **$s5**

163. li $v0, 1

164. syscall

165.

166. end:

167. j exit

168.

169. print\_tab:

170. la $a0, tab\_char

171. li $v0, 4

172. syscall

173. jr $ra

174.

175. print\_end\_line:

176. la $a0, end\_line\_char

177. li $v0, 4

178. syscall

179. jr $ra

180.

181. exit:

* Chú thích code:

|  |  |  |
| --- | --- | --- |
| **Dòng** | **Chú thích** | **Ghi chú** |
| 1 🡪 7 | Khai báo và khởi tạo mảng **array1** có kiểu dữ liệu là word; **array2** có kiểu byte và **array3** kiểu space. Kích thước 3 mảng khác nhau. |  |
| 8 🡪 14 | Khởi tạo các string cần thiết để in ra. |  |
| 21 🡪 23 | In ra thông báo “Xuat mang 1”. |  |
| 24 | In xuống dòng. |  |
| 25 | Load địa chỉ của **array1** vào **$s1**. | **$s1** lưu địa chỉ **array1** |
| 26 | Gán kích thước **array1** vào **$s2**. | **$s2** lưu số phần tử **array1** |
| 28 | Gán **$t0** = 0, để hỗ trợ xử lý duyệt mảng. |  |
| 31 🡪 39 | So sánh **$t0** và **$s2**, thực hiện việc duyệt qua tất cả các phần tử của mảng và in ra các phần tử. |  |
| 41 🡪 42 | In xuống dòng. |  |
| 44 🡪 47 | In ra thông báo “Xuat mang 2”. |  |
| 48 | In xuống dòng. |  |
| 49 | Load địa chỉ của **array2** vào **$s1**. | **$s1** lưu địa chỉ **array2** |
| 50 | Gán kích thước **array2** vào **$s2**. | **$s2** lưu số phần tử **array2** |
| 54 🡪 62 | So sánh **$t0** và **$s2**, thực hiện việc duyệt qua tất cả các phần tử của mảng và in ra các phần tử. |  |
| 64 🡪 65 | In xuống dòng. |  |
| 68 | Load địa chỉ **array2** vào **$s1**. |  |
| 69 🡪 70 | Gán **size2** vào thanh ghi **$s2** và **$s3**. |  |
| 71 🡪 72 | Cộng **$s3** với **$s1**, sau đó trừ đi 1. | Lấy địa chỉ của **array2** cộng với kích thước **array2** rồi gán vào **$s3**, rồi trừ đi 1 |
| 73 | Load địa chỉ **array3** vào **$s4**. |  |
| 74 | Gán kích thước **array3** vào **$s5**. |  |
| 75 | Sao chép địa chỉ của **array3** từ **$s4** sang $s6. |  |
| 76 | Gán **$t0** = 0. |  |
| 77 | Lệnh này sao chép giá trị (địa chỉ đã được điều chỉnh của phần tử cuối cùng trong mảng **array2**) từ thanh ghi **$s3** sang thanh ghi **$t1.** |  |
| 79 🡪80 | So sánh **$t0** và **$s5** để lặp. |  |
| 81 | Lệnh này tính địa chỉ của phần tử hiện tại trong **array2** bằng cách cộng địa chỉ cơ sở của **array2** (**$s1**) với chỉ số hiện tại (**$t0**), và lưu kết quả vào **$t2.** |  |
| 82 🡪 85 | Cộng **$t3** và **$t4** rồi gán vào **$t5**, sau đó store vào **$s6**. | Cộng 2 phần tử đầu và cuối của **array2** rồi store vào **array3** |
| 86 🡪 88 | Tăng và giảm chỉ số ở 2 đầu mảng của **array2**. |  |
| 89 | Jump lại ra **loop\_process\_arr\_3** thực hiện vòng lặp. |  |
| 91 🡪 92 | Xuống dòng. |  |
| 95 🡪 97 | In ra “Xuat mang 3: ”. |  |
| 98 🡪 100 | In xuống dòng.  Load địa chỉ **array3** vào **$s1**.  Load size3 (kích thước mảng 3) vào **$s2**. |  |
| 102 | Gán 0 vào **$t0**. |  |
| 104 🡪 113 | So sánh **$t0** và **$s2** rồi vào vòng lặp.  Thực hiện cộng **$t0** với **$s1** rồi gán vào **$t1**. Load phần tử tại địa chỉ **$t1** vào $a0 rồi in ra.  Sau đó **$t1** tăng thêm 1. Jump về **loop\_print\_arr\_3** để tiếp tục việc lặp. | Duyệt mảng và thực hiện việc in ra tất cả phần tử của mảng. |
| 114 🡪 115 | In xuống dòng. |  |
| 117 🡪 120 | In ra “Nhap thu tu mang can truy xuat: ”. |  |
| 121 🡪 123 | Sao chép chuỗi được nhập vào **$s0**. |  |
| 126 🡪 128 | In ra “Nhap index của mảng: ”. |  |
| 129 🡪 131 | Sao chép chuỗi được nhập vào **$s1**. |  |
| 133 🡪 139 | Thực hiện so sánh chuỗi được nhập. Nếu = 1 thì jump đến **get\_ele\_of\_arr\_1**, nếu = 2 thì jump đến **get\_ele\_of\_arr\_2**, nếu = 3 thì jump đến **get\_ele\_of\_arr\_3**. |  |
| 141 🡪 146 | Thực hiện việc in ra phần tử tại index được nhập vào đối với trường hợp mảng 1. |  |
| 148 🡪152 | Thực hiện việc in ra phần tử tại index được nhập vào đối với trường hợp mảng 2. |  |
| 154 🡪158 | Thực hiện việc in ra phần tử tại index được nhập vào đối với trường hợp mảng 3. |  |
| 160 🡪 164 | Khối lệnh thực hiện việc in phần tử tại index của 1 mảng xác định. |  |
| 169 🡪 173 | Khối lệnh thực hiện in ra kí hiệu ‘**tab**’. |  |
| 175 🡪 179 | Khối lệnh thực hiện in ra kí hiệu xuống dòng. |  |

* Chương trình chạy mẫu:

|  |
| --- |
| xuat mang 1:  5 6 7 8 1 2 3 9 10 4  xuat mang 2:  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16  xuat mang 3:  17 17 17 17 17 17 17 17  Nhap thu tu mang can truy xuat: 2  Nhap index cua mang: 7  8 |
| xuat mang 1:  5 6 7 8 1 2 3 9 10 4  xuat mang 2:  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16  xuat mang 3:  17 17 17 17 17 17 17 17  Nhap thu tu mang can truy xuat: 1  Nhap index cua mang: 4  1 |

|  |
| --- |
| xuat mang 1:  5 6 7 8 1 2 3 9 10 4  xuat mang 2:  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16  xuat mang 3:  17 17 17 17 17 17 17 17  Nhap thu tu mang can truy xuat: 1  Nhap index cua mang: 3  8 |
| xuat mang 1:  5 6 7 8 1 2 3 9 10 4  xuat mang 2:  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16  xuat mang 3:  17 17 17 17 17 17 17 17  Nhap thu tu mang can truy xuat: 2  Nhap index cua mang: 1  2 |

1. **Thao tác với con trỏ**

*(giảm tải)*

1. **Bài tập *(chỉ sử dụng con trỏ)***
   1. **Nhập một mảng các số nguyên n phần tử *(nhập vào số phần tử và giá trị của từng phần tử)*, xuất ra cửa sổ I/O của MARS theo từng yêu cầu sau:**

* Xuất ra giá trị lớn nhất và nhỏ nhất của mảng
* Tổng tất cả các phần tử của mảng
* Người sử dụng nhập vào chỉ số của một phần tử nào đó và giá trị của phần tử
* đó được in ra cửa sổ

1. .data

2.

3. str\_prompt\_in\_num\_of\_eles: .asciiz "Nhap so phan tu cua mang: "

4. str\_prompt\_in\_ele: .asciiz "Nhap phan tu: "

5. str\_promp\_in\_index: .asciiz "Nhap chi so phan tu: "

6. str\_min: .asciiz "Phan tu nho nhat: "

7. str\_max: .asciiz "Phan tu lon nhat: "

8. str\_sum: .asciiz "Tong cua mang: "

9. str\_index\_value: .asciiz "Phan tu tai index vua nhap: "

10. end\_line\_char: .asciiz "\n"

11. array: .space 100

12.

13. .text

14.

15. main:

16.

17. input\_array:

18. li $v0, 4

19. la $a0, str\_prompt\_in\_num\_of\_eles

20. syscall

21. li $v0, 5

22. syscall

23. move **$s0**, $v0

24. la **$s1**, array

25.

26. read\_array:

27. li **$t0**, 0

28.

29. loop\_read\_array:

30. bge **$t0**, **$s0**, prepare\_min

31. la $a0, str\_prompt\_in\_ele

32. li $v0, 4

33. syscall

34. li $v0, 5

35. syscall

36. add **$t1**, **$s1**, **$t0**

37. sb $v0, 0(**$t1**)

38. addi **$t0**, **$t0**, 1

39. j loop\_read\_array

40.

41. prepare\_min:

42. li **$t0**, 1

43. lb **$s2**, 0(**$s1**)

44.

45. loop\_get\_min:

46. bge **$t0**, **$s0**, print\_min

47. add **$t1**, **$s1**, **$t0**

48. addi **$t0**, **$t0**, 1

49. lb $t2, 0(**$t1**)

50. bge $t2, **$s2**, loop\_get\_min

51. move **$s2**, $t2

52. j loop\_get\_min

53.

54. print\_min:

55. li $v0, 4

56. la $a0, str\_min

57. syscall

58. move $a0, **$s2**

59. li $v0, 1

60. syscall

61.

62. jal print\_end\_line

63.

64. prepare\_max:

65. li **$t0**, 1

66. lb **$s3**, 0(**$s1**)

67.

68. loop\_get\_max:

69. bge **$t0**, **$s0**, print\_max

70. add **$t1**, **$s1**, **$t0**

71. addi **$t0**, **$t0**, 1

72. lb $t2, 0(**$t1**)

73. ble $t2, **$s3**, loop\_get\_max

74. move **$s3**, $t2

75. j loop\_get\_max

76.

77. print\_max:

78. li $v0, 4

79. la $a0, str\_max

80. syscall

81. move $a0, **$s3**

82. li $v0, 1

83. syscall

84.

85. jal print\_end\_line

86.

87. prepare\_sum:

88. li **$t0**, 0

89. li **$s4**, 0

90.

91. loop\_get\_sum:

92. bge **$t0**, **$s0**, print\_sum

93. add **$t1**, **$s1**, **$t0**

94. lb $t2, 0(**$t1**)

95. add **$s4**, **$s4**, $t2

96. addi **$t0**, **$t0**, 1

97. j loop\_get\_sum

98.

99. print\_sum:

100. li $v0, 4

101. la $a0, str\_sum

102. syscall

103. move $a0, **$s4**

104. li $v0, 1

105. syscall

106.

107. jal print\_end\_line

108.

109. input\_index:

110. la $a0, str\_promp\_in\_index

111. li $v0, 4

112. syscall

113. li $v0, 5

114. syscall

115. move **$t0**, $v0

116.

117. print\_index\_value:

118. la $a0, str\_index\_value

119. li $v0, 4

120. syscall

121. add **$t1**, **$t0**, **$s1**

122. lb $a0, 0(**$t1**)

123. li $v0, 1

124. syscall

125.

126. end:

127. j exit

128.

129. print\_end\_line:

130. la $a0, end\_line\_char

131. li $v0, 4

132. syscall

133. jr $ra

134.

135. exit:

* Chú thích code:

|  |  |  |
| --- | --- | --- |
| **Dòng** | **Chú thích** | **Ghi chú** |
| 3 🡪 10 | Các string cần thiết để xuất ra màn hình. |  |
| 11 | Khởi tạo **array** với kiểu **space** có kích thước 100 bytes. |  |
| 18 🡪 20 | Xuất thông báo yêu cầu nhập số phần tử của **array**. |  |
| 21 🡪 23 | Nhập số phần tử của **array** và lưu vào **$s0**. | **$s0** lưu số phần tử. |
| 24 | Load địa chỉ của **array** vào **$s1**. | **$s1** lưu địa chỉ array. |
| 27 | Gán **$t0** = 0nhằm hỗ trợ việc lặp xử lý mảng. |  |
| 30 | So sánh nếu **$t0**  **≥ $s0** thì jump label **prepare\_min**. |  |
| 31 🡪 35 | Xuất nhập phần tử lần lượt. |  |
| 36 🡪 38 | Lưu giá trị vừa nhập vào index tương ứng lần lượt vào **array**. | Nhập giá trị cho các phần tử của **array**. |
| 39 | Jump ngược lên dòng 30 để lặp. |  |
| 42 🡪 43 | Gán **$t0** = 1nhằm hỗ trợ việc lặp xử lý mảng.  Gán **$s2** là phần tử đầu tiên của mảng. | **$s2** là giá trị của phần tử nhỏ nhất. |
| 46 🡪 52 | Việc lặp để kiểm tra xem có phần tử nào nhỏ hơn giá trị hiện tại của **$s2** không. Nếu có thì gán giá trị đó vào **$s2**. |  |
| 55 🡪 60 | Xuất ra giá trị nhỏ nhất của **array**. | Xuất **$s2**. |
| 65 🡪 66 | Gán **$t0** = 1nhằm hỗ trợ việc lặp xử lý mảng.  Gán **$s3** là phần tử đầu tiên của mảng. | **$s3** là giá trị của phần tử lớn nhất của mảng. |
| 69 🡪 75 | Việc lặp để kiểm tra xem có phần tử nào lớn hơn giá trị hiện tại của **$s3** không. Nếu có thì gán giá trị đó vào **$s3**. |  |
| 77 🡪 83 | Xuất ra giá trị lớn nhất của **array**. | Xuất **$s3**. |
| 88 🡪 89 | Gán **$t0** = 0nhằm hỗ trợ việc lặp xử lý mảng.  Gán **$s4** = 0. | **$s4** là tổng giá trị các phần tử của **array**. |
| 92 🡪 97 | Lặp cộng dần từng phần tử của **array** vào **$s4**. |  |
| 100 🡪 105 | Xuất ra tổng các phần tử của **array**. | Xuất **$s4**. |
| 110 🡪 115 | Xuất và yêu cầu nhập index muốn truy xuất từ **array**. Lưu giá trị index vừa nhập vào **$t0**. | **$t0** là giá trị index muốn truy xuất. |
| 118 🡪 124 | Xuất ra giá trị của phần tử index thứ **$t0**. | Xuất giá trị phần tử tại index **$t0**. |

* Chương trình chạy mẫu:

|  |
| --- |
| Nhap so phan tu cua mang: 7  Nhap phan tu: 2  Nhap phan tu: 1  Nhap phan tu: 5  Nhap phan tu: 4  Nhap phan tu: 3  Nhap phan tu: 4  Nhap phan tu: 7  Phan tu nho nhat: 1  Phan tu lon nhat: 7  Tong cua mang: 26  Nhap chi so phan tu: 3  Phan tu tai index vua nhap: 4 |
| Nhap so phan tu cua mang: 5  Nhap phan tu: 7  Nhap phan tu: 2  Nhap phan tu: 5  Nhap phan tu: 4  Nhap phan tu: 9  Phan tu nho nhat: 2  Phan tu lon nhat: 9  Tong cua mang: 27  Nhap chi so phan tu: 2  Phan tu tai index vua nhap: 5 |

* 1. **Nhập một mảng các số nguyên n phần tử *(nhập vào số phần tử và giá trị của từng phần tử)*. Mảng này gọi là A.**

Chuyển dòng lệnh C dưới đây sang mã assembly của MIPS. Với các biến nguyên i, j được gán lần lượt vào thanh ghi **$s0**, **$s1**; và địa chỉ nền của mảng số nguyên A được lưu trong thanh ghi **$s3**.

1. if (i<j) A[i]= i;

2. else A[i] = j;

* Bài code:

1. .data

2.

3. str\_prompt\_in\_1: .asciiz "Nhap so phan tu cua mang: "

4. str\_prompt\_in\_2: .asciiz "Nhap phan tu: "

5. str\_prompt\_in\_i: .asciiz "Nhap i: "

6. str\_prompt\_in\_j: .asciiz "Nhap j: "

7. tab\_char: .asciiz "\t"

8. array: .space 100

9.

10. .text

11.

12. main:

13.

14. read\_number\_of\_eles\_of\_arr:

15. li $v0, 4

16. la $a0, str\_prompt\_in\_1

17. syscall

18. li $v0, 5

19. syscall

20. move **$t0**, $v0

21. la $a3, array

22. li **$t1**, 0

23.

24. read\_array:

25. bge **$t1**, **$t0**, input\_i\_j

26. li $v0, 4

27. la $a0, str\_prompt\_in\_2

28. syscall

29. li $v0, 5

30. syscall

31. add $t5, $a3, **$t1**

32. sb $v0, 0($t5)

33. addi **$t1**, **$t1**, 1

34. j read\_array

35.

36. input\_i\_j:

37. la **$s3**, array

38. li $v0, 4

39. la $a0, str\_prompt\_in\_i

40. syscall

41. li $v0, 5

42. syscall

43. move **$s0**, $v0

44. li $v0, 4

45. la $a0, str\_prompt\_in\_j

46. syscall

47. li $v0, 5

48. syscall

49. move **$s1**, $v0

50.

51. compare\_i\_j:

52. add $t3, **$s3**, **$s0**

53. blt **$s0**, **$s1**, asign\_i

54. j asign\_j

55.

56. asign\_i:

57. sb **$s0**, 0($t3)

58. j continue\_2

59.

60. asign\_j:

61. sb **$s1**, 0($t3)

62.

63. continue\_2:

64. la $a3, array

65. li **$t1**, 0

66.

67. print:

68. bge **$t1**, **$t0**, end

69. add $t5, $a3, **$t1**

70. lb $a0, 0($t5)

71. li $v0, 1

72. syscall

73. jal print\_tab

74. addi **$t1**, **$t1**, 1

75. j print

76.

77. end:

78. j exit

79.

80. print\_tab:

81. la $a0, tab\_char

82. li $v0, 4

83. syscall

84. jr $ra

85.

86. exit:

* Chú thích code:

|  |  |  |
| --- | --- | --- |
| **Dòng** | **Chú thích** | **Ghi chú** |
| 3 🡪 7 | Các string cần thiết để xuất ra màn hình. |  |
| 8 | Khởi tạo **array** với kiểu **space** có kích thước 100 bytes. |  |
| 15 🡪 17 | Xuất thông báo yêu cầu nhập số phần tử của **array**. |  |
| 18 🡪 20 | Nhập số phần tử của **array** và lưu vào **$t0**. | **$t0** lưu số phần tử. |
| 21 | Load địa chỉ của **array** vào **$a3**. | **$a3** lưu địa chỉ array. |
| 22 | Gán **$t1** = 0nhằm hỗ trợ việc lặp xử lý mảng. |  |
| 25 🡪 34 | Xuất và yêu cầu nhập giá trị cho từng phần tử vào **array**. |  |
| 37 🡪 43 | Nhập i và lưu vào **$s0**. | **$s0** là i. |
| 44 🡪 49 | Nhập j và lưu vào **$s1**. | **$s1** là j. |
| 52 | Lưu địa chỉ của array index thứ i vào **$t3** | **$t3** là vị tí cần thay thế |
| 53 | So sánh i và j. Nếu i < j thì **array[i]** = i tại label **asign\_i**, ngược lại **array[i]** = j tại label **asign\_j**. | Gán **array[i]** theo yêu cầu của đề bài. |
| 64 🡪 65 | Lưu địa chỉ của **array** vào **$a3**.  Gán **$t1** = 0 để hỗ trợ lặp xuất **array**. |  |
| 68 🡪 75 | Lặp xuất **array**. |  |

* Chương trình chạy mẫu:
  + TH i < j:

|  |
| --- |
| Nhap so phan tu cua mang: 7  Nhap phan tu: 3  Nhap phan tu: 4  Nhap phan tu: 1  Nhap phan tu: 2  Nhap phan tu: 5  Nhap phan tu: 7  Nhap phan tu: 6  Nhap i: 2  Nhap j: 6  3 4 2 2 5 7 6 |
| Nhap so phan tu cua mang: 5  Nhap phan tu: 3  Nhap phan tu: 4  Nhap phan tu: 6  Nhap phan tu: 1  Nhap phan tu: 3  Nhap i: 1  Nhap j: 2  3 1 6 1 3 |

* TH i > j:

|  |
| --- |
| Nhap so phan tu cua mang: 7  Nhap phan tu: 3  Nhap phan tu: 4  Nhap phan tu: 1  Nhap phan tu: 2  Nhap phan tu: 6  Nhap phan tu: 7  Nhap phan tu: 6  Nhap i: 4  Nhap j: 1  3 4 1 2 1 7 6 |
| Nhap so phan tu cua mang: 5  Nhap phan tu: 3  Nhap phan tu: 4  Nhap phan tu: 6  Nhap phan tu: 1  Nhap phan tu: 3  Nhap i: 4  Nhap j: 2  3 4 6 1 2 |