**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 04**

**MỤC LỤC**

[1. Lý thuyết 3](#_Toc165848014)

[2. Thực hành 3](#_Toc165848015)

[2.1. Bài 1 3](#_Toc165848016)

[2.2. Bài 2 4](#_Toc165848017)

[3. Bài tập 5](#_Toc165848018)

[3.1. Nhập vào một ký tự, xuất ra cửa sổ I/O của MARS theo từng yêu cầu 5](#_Toc165848019)

[3.2.1. Nhập 2 số nguyên, in ra cửa sổ I/O của MARS theo từng yêu cầu 8](#_Toc165848020)

[3.2.2. Nhập 2 số thực, in ra cửa sổ I/O của MARS theo từng yêu cầu 12](#_Toc165848021)

[4. Bài tập về nhà 15](#_Toc165848022)

[4.1. Nhập 1 chuỗi, xuất ra chuỗi đảo ngược 15](#_Toc165848023)

[4.2. Nhập 1 mảng số nguyên và sắp xếp theo thứ tự tăng dần 17](#_Toc165848024)

1. **Lý thuyết**

* Giảng viên hướng dẫn sinh viên về chương trình hợp ngữ MIPS dựa theo tài liệu: **Tổng quát về hợp ngữ và kiến trúc MIPS**

1. **Thực hành**
   1. **Bài 1**

1. main:

2. # nhap i

3. li $v0, 5

4. syscall

5. add $s0, $v0, $zero

6.

7. # nhap j

8. li $v0, 5

9. syscall

10. add $s1, $v0, $zero

11.

12. # nhap g

13. li $v0, 5

14. syscall

15. add $t0, $v0, $zero

16.

17. # nhap h

18. li $v0, 5

19. syscall

20. add $t1, $v0, $zero

21.

22. # so sanh

23. beq $s0, $s1, if

24. sub $s2, $t0, $t1

25.

26. j continue

27.

28. if:

29. add $s2, $t0, $t1

30.

31. # in ket qua

32. continue:

33. li $v0, 1

34. add $a0, $s2, $zero

35. syscall

**Chương trình mẫu:**

A screenshot of a computer

Description automatically generated

|  |  |  |
| --- | --- | --- |
| **Dòng** | **Chú thích** | **Ghi chú** |
| 3 🡪 5 | Nhập i và lưu vào register **$s0**. |  |
| 8 🡪 10 | Nhập j và lưu vào register **$s1**. |  |
| 12 🡪 15 | Nhập g và lưu vào register **$t0**. |  |
| 18 🡪 20 | Nhập h và lưu vào register **$t1**. |  |
| 23 | So sánh **$s0** và **$s1.**  **N**ếu **$s0 = $s1** thực hiện tiếp dòng 28, ngược lại thì thực hiện dòng tiếp theo. | So sánh i và j. |
| 24 | Lấy **$t0 - $t1** rồi lưu vào register **$s2**. | Lấy g – h rồi lưu vào **$s2** |
| 28 🡪 29 | Lấy **$t0 + $t1** rồi lưu vào register **$s2**. | Lấy g + h rồi lưu vào **$s2** |
| 32 🡪 35 | In kết quả đã tính toán. |  |

* 1. **Bài 2**

1. .data

2. i: .word 1

3. Sum: .word 0

4.

5. .text

6. main:

7.

8. # gan

9. lw $s0, i

10. lw $s2, Sum

11.

12. # nhap N

13. li $v0, 5

14. syscall

15. add $s1, $v0, $zero

16.

17. loop: bgt $s0, $s1, continue

18. add $s2, $s2, $s0

19. addi $s0, $s0, 1

20. j loop

21.

22. continue:

23. li $v0, 1

24. add $a0, $s2, $zero

25. syscall

**Chương trình mẫu:**

A screen shot of a computer

Description automatically generated

|  |  |  |
| --- | --- | --- |
| **Dòng** | **Chú thích** | **Ghi chú** |
| 2 | Lưu i = 1**.** |  |
| 3 | Lưu Sum = 0**.** |  |
| 9 | Lưu i vào register **$s0**. |  |
| 10 | Lưu Sum vào register **$s2**. |  |
| 13 🡪 15 | Nhập N và lưu vào register **$s1**. |  |
| 17 🡪 19 | So sánh **$s0** và **$s1** Nếu **$s0 > $s1**, thực hiện tiếp dòng 22, ngược lại thực hiện dòng tiếp theo đến khi gặp dòng lệnh “**j loop**” thì quay lại dòng 17.  Cộng **$s2 và $s0** rồi lưu vào **$s2**.  Cộng **$s0 với 1** rồi lưu vào **$s0**. | Thực hiện cộng Sum và I trong vòng lặp với điều kiện i <= N, i++. |
| 22 🡪 25 | In ra **$s2**, | In ra Sum sau khi kết thúc vòng lặp. |

1. **Bài tập**
   1. **Nhập vào một ký tự, xuất ra cửa sổ I/O của MARS theo từng yêu cầu**
      * Ký tự liền trước và liền sau của ký tự nhập vào
      * Ký tự nhập vào chỉ được phép là ba loại: số, chữ thường và chữ hoa. Nếu ký tự nhập vào rơi vào một trong ba loại, xuất ra cửa sổ đó là loại nào; nếu ký tự nhập không rơi vào một trong ba loại trên, xuất ra thông báo “invalid type”

1. .data

2. string\_truoc: .asciiz "Ky tu truoc: "

3. string\_sau: .asciiz "Ky tu sau: "

4. string\_invalid: .asciiz "invalid type"

5. string\_xuong\_dong: .asciiz "\n"

6.

7. .text

8. main:

9. # read character

10. li $v0, 12

11. syscall

12. add $s0, $v0, $zero

13. jal xuong\_dong

14.

15. la\_so:

16. li $t0, 48

17. blt $s0, $t0, invalid

18. li $t0, 57

19. ble $s0, $t0, get

20.

21. la\_chu\_hoa:

22. li $t0, 65

23. blt $s0, $t0, invalid

24. li $t0, 90

25. ble $s0, $t0, get

26.

27. la\_chu\_thuong:

28. li $t0, 97

29. blt $s0, $t0, invalid

30. li $t0, 122

31. ble $s0, $t0, get

32. j invalid

33.

34. get:

35. # get truoc va sau

36. addi $s1, $s0, -1

37. addi $s2, $s0, 1

38.

39. valid:

40. # in ra truoc

41. li $v0, 4

42. la $a0, string\_truoc

43. syscall

44. li $v0, 11

45. add $a0, $s1, $zero

46. syscall

47. jal xuong\_dong

48. # in ra sau

49. li $v0, 4

50. la $a0, string\_sau

51. syscall

52. li $v0, 11

53. add $a0, $s2, $zero

54. syscall

55. jal xuong\_dong

56. j end

57.

58. invalid:

59. # invalid

60. li $v0, 4

61. la $a0, string\_invalid

62. syscall

63. j end

64.

65. xuong\_dong:

66. li $v0, 4

67. la $a0, string\_xuong\_dong

68. syscall

69. jr $ra

70.

71. end:

**Chương trình mẫu**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

|  |  |  |
| --- | --- | --- |
| **Dòng** | **Chú thích** | **Ghi chú** |
| 2 🡪 5 | Lưu các biến **string\_truoc** = "Ky tu truoc: "  **string\_sau** = "Ky tu sau: "  **string\_invalid** = "invalid type"  **string\_xuong\_dong**  = "\n" |  |
| 10 🡪 12 | Nhập kí tự và lưu vào $s0. |  |
| 15 🡪 19 | So sánh mã ascii của **$s0** với mã ascii 48 (số 0). Nếu nhỏ hơn 48 thì nhảy đến label **invalid.** Ngược lại tiếp tục thực hiện dòng tiếp theo, so sánh với mã ascii 57 (số 9), nếu nhỏ hơn hoặc bằng 57 thì nhảy đến label **get.** | Khối điều kiện kiểm tra kí tự nhập vào có phải là số hay không. |
| 21 🡪 25 | So sánh mã ascii của **$s0** với mã ascii 65 (“A”). Nếu nhỏ hơn 65 thì nhảy đến label **invalid.** Ngược lại tiếp tục thực hiện dòng tiếp theo, so sánh với mã ascii 90 (“Z”), nếu nhỏ hơn hoặc bằng 90 thì nhảy đến label **get**. | Khối điều kiện kiểm tra kí tự nhập vào có phải là chữ hoa hay không. |
| 27 🡪 31 | So sánh mã ascii của **$s0** với mã ascii 97 (“a”). Nếu nhỏ hơn 97 thì nhảy đến **label invalid.** Ngược lại tiếp tục thực hiện dòng tiếp theo, so sánh với mã ascii 122 (“z”), nếu nhỏ hơn hoặc bằng 122 thì nhảy đến label **get**. | Khối điều kiện kiểm tra kí tự nhập vào có phải là chữ thường hay không. |
| 32 | Nhảy đến label **invalid**. |  |
| 34 🡪 37 | Lần lượt bớt và thêm 1 cho mã ascii của **$s0** rồi lưu vào **$s1 và $s2**. | Gán kí tự trước và sau kí tự đã nhập vào register **$s1, $s2**. |
| 39 🡪 56 | In ra màn hình như ví dụ sau:  “b  Ky tu truoc: a  Ky tu sau: c” | Label **valid** |
| 58 🡪 63 | In ra thông báo “invalid type”. | Label **invalid** |
| 65 🡪 69 | In xuống dòng. |  |
| 71 | Kết thúc. |  |

**3.2.1. Nhập 2 số nguyên, in ra cửa sổ I/O của MARS theo từng yêu cầu**

* + - Số lớn hơn
    - Tổng, hiệu, tích, thương của 2 số

1. .data

2. string\_nhap\_a: .asciiz "Nhap a: "

3. string\_nhap\_b: .asciiz "Nhap b: "

4. string\_tong: .asciiz "a + b = "

5. string\_hieu: .asciiz "a - b = "

6. string\_nhan: .asciiz "a \* b = "

7. string\_khong\_the\_chia: .asciiz "a khong the chia cho b!"

8. string\_chia: .asciiz "a / b = "

9. string\_du: .asciiz "du = "

10. string\_soLon: .asciiz "So lon hon la: "

11. string\_xuong\_dong: .asciiz "\n"

12.

13. .text

14.

15. main:

16. # nhap a

17. la $a0, string\_nhap\_a

18. li $v0, 4

19. syscall

20. li $v0, 5

21. syscall

22. add $s0, $v0, $zero

23.

24. # nhap b

25. la $a0, string\_nhap\_b

26. li $v0, 4

27. syscall

28. li $v0, 5

29. syscall

30. add $s1, $v0, $zero

31.

32. jal xuat\_so\_lon

33. slt $t1, $s0, $s1

34. beqz $t1, soLon

35. add $a0, $s1, $zero

36. li $v0, 1

37. syscall

38. jal xuong\_dong

39. j tong

40.

41. soLon:

42. add $a0, $s0, $zero

43. li $v0, 1

44. syscall

45. jal xuong\_dong

46.

47. tong:

48.

49. # xuat tinh tong

50. la $a0, string\_tong

51. li $v0, 4

52. syscall

53.

54. # xuat gia tri tinh tong

55. add $a0, $s0, $s1

56. li $v0, 1

57. syscall

58. jal xuong\_dong

59.

60. hieu:

61.

62. # xuat tinh hieu

63. la $a0, string\_hieu

64. li $v0, 4

65. syscall

66.

67. # xuat gia tri tinh hieu

68. sub $a0, $s0, $s1

69. li $v0, 1

70. syscall

71. jal xuong\_dong

72.

73. nhan:

74.

75. # xuat tinh nhan

76. la $a0, string\_nhan

77. li $v0, 4

78. syscall

79.

80. # xuat gia tri tinh nhan

81. mult $s0, $s1

82. mflo $a0

83. li $v0, 1

84. syscall

85. jal xuong\_dong

86.

87. chia:

88.

89. # kiem tra b == 0

90. beq $s1, $zero, khong\_the\_chia

91.

92. # xuat tinh chia

93. la $a0, string\_chia

94. li $v0, 4

95. syscall

96.

97. # xuat gia tri thuong

98. div $s0, $s1

99. mflo $a0

100. li $v0, 1

101. syscall

102. jal xuong\_dong

103.

104. # so du

105. mfhi $t0

106. beq $t0, $zero, end

107. la $a0, string\_du

108. li $v0, 4

109. syscall

110. add $a0, $t0, $zero

111. li $v0, 1

112. syscall

113. j end

114.

115. khong\_the\_chia:

116.

117. # xuat khong the chia

118. la $a0, string\_khong\_the\_chia

119. li $v0, 4

120. syscall

121. j end

122.

123. xuat\_so\_lon:

124.

125. li $v0, 4

126. la $a0, string\_soLon

127. syscall

128. jr $ra

129.

130. xuong\_dong:

131. li $v0, 4

132. la $a0, string\_xuong\_dong

133. syscall

134. jr $ra

135.

136. end:

A computer screen with a blue and white text

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated

|  |  |  |
| --- | --- | --- |
| **Dòng** | **Chú thích** | **Ghi chú** |
| 2 🡪 11 | Lưu các biến string cần thiết để xuất trong quá trình chạy chương trình. |  |
| 17🡪 19 | Xuất yêu cầu nhập a. |  |
| 20 🡪 22 | Nhập a và lưu giá trị vào register **$s0**. |  |
| 25 🡪 27 | Xuất yêu cầu nhập b. |  |
| 28 🡪 30 | Nhập b và lưu giá trị vào register **$s1**. |  |
| 32 | Xuất “Số lớn hơn là: ”. |  |
| 33 🡪 45 | So sánh **$s0** và **$s1**, sau đó xuất ra số lớn hơn. | So sánh a và b |
| 50 🡪 52 | Xuất tính tổng. |  |
| 55 🡪 57 | Tính tổng của **$s0** và **$s1** lưu vào **$a0**. Sau đó xuất ra màn hình giá trị **$a0**. | a + b |
| 58 | Xuất xuống dòng. |  |
| 63 🡪 65 | Xuất tính hiệu. |  |
| 68 🡪 70 | Tính hiệu của **$s0** và **$s1** lưu vào **$a0**. Sau đó xuất ra màn hình giá trị **$a0**. | a – b |
| 71 | Xuất xuống dòng. |  |
| 76 🡪 78 | Xuất tính nhân. |  |
| 81 🡪 84 | Tính nhân *(tích)* của **$s0** và **$s1** lưu vào **$a0**. Sau đó xuất ra màn hình giá trị **$a0**. | a \* b |
| 85 | Xuất xuống dòng. |  |
| 90 | Kiểm tra xem **$s1** có bằng **$zero**?  Nếu bằng thì jump đến label **khong\_the\_chia** và jump đến label **end** kết thúc chương trình. | b == 0 ? |
| 93 🡪 95 | Xuất tính chia. |  |
| 98 🡪 101 | Tính chia *(thương)* **$s0** chia **$s1** và lưu kết quả vào **$a0**. Sau đó xuất ra màn hình giá tị **$a0**. | a / b |
| 102 | Xuất xuống dòng. |  |
| 105 | Lưu số dư của kêt quả phép chia trên vào **$t0.** |  |
| 106 | Kiểm tra **$t0** có bằng 0?  Nếu bằng thì jump đến label **end** và kết thúc chương trình.  Nếu không bằng thì tiếp tục. | Số dư == 0 ? |
| 107🡪 109 | Xuất tính dư. |  |
| 110 🡪 113 | Lưu giá trị **$t0** vào **$a0** và xuất ra màn hình giá trị **$a0**. Jump đến label **end** và kết thúc chương trình. | Xuất số dư. |

**3.2.2. Nhập 2 số thực, in ra cửa sổ I/O của MARS theo từng yêu cầu**

* + - Số lớn hơn
    - Tổng, hiệu, tích, thương của 2 số

1. .data

2. string\_nhap\_a: .asciiz "Nhap a: "

3. string\_nhap\_b: .asciiz "Nhap b: "

4. string\_tong: .asciiz "a + b = "

5. string\_hieu: .asciiz "a - b = "

6. string\_nhan: .asciiz "a \* b = "

7. string\_khong\_the\_chia: .asciiz "a khong the chia cho b!"

8. string\_chia: .asciiz "a / b = "

9. string\_xuong\_dong: .asciiz "\n"

10.

11. .text

12. main:

13. # nhap a

14. la $a0, string\_nhap\_a

15. li $v0, 4

16. syscall

17. li $v0, 6

18. syscall

19. mov.s $f1, $f0

20.

21. # nhap b

22. la $a0, string\_nhap\_b

23. li $v0, 4

24. syscall

25. li $v0, 6

26. syscall

27. mov.s $f2, $f0

28.

29. tong:

30. # xuat tinh tong

31. la $a0, string\_tong

32. li $v0, 4

33. syscall

34. # xuat gia tri tinh tong

35. add.s $f12, $f1, $f2

36. li $v0, 2

37. syscall

38. jal xuong\_dong

39.

40. hieu:

41. # xuat tinh hieu

42. la $a0, string\_hieu

43. li $v0, 4

44. syscall

45. # xuat gia tri tinh hieu

46. sub.s $f12, $f1, $f2

47. li $v0, 2

48. syscall

49. jal xuong\_dong

50.

51. nhan:

52. # xuat tinh nhan

53. la $a0, string\_nhan

54. li $v0, 4

55. syscall

56. # xuat gia tri tinh nhan

57. mul.s $f12, $f1, $f2

58. li $v0, 2

59. syscall

60. jal xuong\_dong

61.

62. chia:

63. # kiem tra b == 0

64. mtc1 $zero, $f0

65. c.eq.s $f2, $f0

66. bc1t khong\_the\_chia

67. # xuat tinh chia

68. la $a0, string\_chia

69. li $v0, 4

70. syscall

71. # xuat gia tri thuong

72. div.s $f12, $f1, $f2

73. li $v0, 2

74. syscall

75. j end

76.

77. khong\_the\_chia:

78. # xuat khong the chia

79. la $a0, string\_khong\_the\_chia

80. li $v0, 4

81. syscall

82. j end

83.

84. xuong\_dong:

85. li $v0, 4

86. la $a0, string\_xuong\_dong

87. syscall

88. jr $ra

89.

90. end:

**Chương trình mẫu:**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

|  |  |  |
| --- | --- | --- |
| **Dòng** | **Chú thích** | **Ghi chú** |
| 2 🡪 9 | Lưu các biến string cần thiết để xuất trong quá trình chạy chương trình. |  |
| 14 🡪 16 | Xuất yêu cầu nhập a. |  |
| 17 🡪 19 | Nhập a và lưu giá trị vào register **$f1**. |  |
| 22 🡪 24 | Xuất yêu cầu nhập b. |  |
| 25 🡪 27 | Nhập b và lưu giá trị vào register **$f2**. |  |
| 31 🡪 33 | Xuất tính tổng. |  |
| 35 🡪 37 | Tính tổng của **$f1** và **$f2** lưu vào **$f12**. Sau đó xuất ra màn hình giá trị **$f12**. | a + b |
| 38 | Xuất xuống dòng. |  |
| 42 🡪 44 | Xuất tính hiệu. |  |
| 46 🡪 48 | Tính hiệu của **$f1** và **$f2** lưu vào **$12**. Sau đó xuất ra màn hình giá trị **$f12**. | a – b |
| 49 | Xuất xuống dòng. |  |
| 53 🡪 55 | Xuất tính nhân. |  |
| 57 🡪 59 | Tính nhân *(tích)* của **$f1** và **$f2** lưu vào **$f12**. Sau đó xuất ra màn hình giá trị **$f12**. | a \* b |
| 60 | Xuất xuống dòng. |  |
| 64 | Gán giá trị 0 của **$zero** vào register **$f0** |  |
| 65 | So sánh giá trị của **$f2** có bằng 0 *(giá trị của $f0)*?  Đúng sẽ gán giá trị 1 cho flag, sai sẽ gán giá trị 0. | b == 0 ? |
| 66 | Nếu flag bằng 1 thì jump đến label **khong\_the\_chia**. Sau đó jump đến label **end** và kết thúc chương trình.  Nếu flag bằng 0 thì tiếp tục chương trình. |  |
| 68 🡪 70 | Xuất tính chia. |  |
| 72 🡪 74 | Tính chia *(thương)* **$f1** chia **$f2** và lưu kết quả vào **$f12**. Sau đó xuất ra màn hình giá tị **$f12**. | a / b |
| 75 | Jump đến label **end** và kết thúc chương trình. |  |

1. **Bài tập về nhà**
   1. **Nhập 1 chuỗi, xuất ra chuỗi đảo ngược**

1. .data

2. string\_nhap: .asciiz "Nhap chuoi: "

3. string\_xuat: .asciiz "Chuoi dao nguoc: "

4. string: .space 100

5.

6. .text

7. main:

8. nhap:

9. li $v0, 4

10. la $a0, string\_nhap

11. syscall

12.

13. li $v0, 8

14. la $a0, string

15. li $a1, 100

16. syscall

17. la $t1, string

18. add $t1, $t1, $a1

19. addi $t2, $a1, 0

20.

21. xuat:

22. li $v0, 4

23. la $a0, string\_xuat

24. syscall

25.

26. xuat\_nguoc:

27. slti $s1, $t2, 0

28. bnez $s1, end

29. lb $a0, 0($t1)

30. li $v0, 11

31. syscall

32. addi $t2, $t2, -1

33. addi $t1, $t1, -1

34. j xuat\_nguoc

35. end:

**Chương trình mẫu:**

A screenshot of a computer

Description automatically generated

|  |  |  |
| --- | --- | --- |
| **Dòng** | **Chú thích** | **Ghi chú** |
| 2 🡪 3 | Lưu các biến string cần thiết để xuất trong quá trình chạy chương trình. |  |
| 4 | Label **string** là kiểu **space** với kích thước 100. | = 100 bytes |
| 9 🡪 11 | Xuất yêu cầu nhập chuỗi. |  |
| 13 🡪 19 | Nhập chuỗi độ dài tối đa = 100, **$a1 = 100**  Lưu địa chỉ kí tự đầu tiên của chuỗi vào **$t1**  Lưu địa chỉ kí tự cuối của chuỗi vào **$t1** bằng cách cho **$t1 + $a1** Lưu **$t2 = $a1** |  |
| 21 🡪 24 | Xuất chuỗi đã nhập |  |
| 26 🡪 27 | So sánh **$t2** với **0** rồi lưu vào **$s1 ( $s1 = 1** nếu **$t2 < 0** và ngược lại **$s1 = 0).** | Thực hiện in ngược chuỗi đã nhập với điều kiện **$t2 >= 0, $t2 --.** |
| 28 | Lệnh nhảy tới **Label end** nếu giá trị trong **$s1** khác 0, tức là nếu **$t2** nhỏ hơn 0. |
| 29 🡪 31 | In kí tự tại địa chỉ **$t1** |
| 32 🡪 33 | Giảm **$t2** và **$t1** |
| 34 | Nhảy về thực hiện lại từ dòng 26 |

* 1. **Nhập 1 mảng số nguyên và sắp xếp theo thứ tự tăng dần**