

IT3280

THỰC HÀNH KIẾN TRÚC MÁY TÍNH

BÁO CÁO MINI-PROJECT

Sinh viên 1:

Nguyễn Trung THÀNH
20176874

Sinh viên 2:

Hoàng Thị Thu TRANG
20176891

Giảng viên hướng dẫn:

TS. Lê Xuân THÀNH

Ngày 18 tháng 5 năm 2020

Mục lục

1	Tổng Quan	2
1.1	Đề tài được phân công	2
1.2	Đề tài làm thêm	2
1.3	Công cụ sử dụng	3
1.4	Source Code	3
2	Project 7 – Sort By Height	4
2.1	Phân tích cách thức thực hiện	4
2.2	Ý nghĩa của các thanh ghi được sử dụng	4
2.3	Source code	5
2.4	Kết quả chạy chương trình	8
3	Project 8 – Students	9
3.1	Phân tích cách thức thực hiện	9
3.2	Ý nghĩa của các thanh ghi được sử dụng	9
3.3	Ý nghĩa của chương trình con	10
3.4	Source code	10
3.5	Kết quả chạy chương trình	15
4	Project 21 – Digit Degree	16
4.1	Phân tích cách thức thực hiện	16
4.2	Ý nghĩa của các thanh ghi được sử dụng	16
4.3	Ý nghĩa của chương trình con	16
4.4	Source code	16
4.5	Kết quả chạy chương trình	19
5	Project 23 – Surpassing Words	20
5.1	Phân tích cách thức thực hiện	20
5.2	Ý nghĩa của các thanh ghi được sử dụng	20
5.3	Ý nghĩa của chương trình con	21
5.4	Source code	21
5.5	Kết quả chạy chương trình	24
6	Project 24 – Cyclone Word (Challenge)	25
6.1	Phân tích cách thức thực hiện	25
6.2	Ý nghĩa của các thanh ghi được sử dụng	25
6.3	Ý nghĩa của chương trình con	25
6.4	Source code	26
6.5	Kết quả chạy chương trình	28

1 Tổng Quan

1.1 Đề tài được phân công

1. Project 7 - Sort By Height

Some people are standing in a row in a park. There are trees between them which cannot be moved. Your task is to rearrange the people by their heights in a nondescending order without moving the trees. People can be very tall!

Example: For $a = [-1, 150, 190, 170, -1, -1, 160, 180]$, the output should be $\text{sortByHeight}(a) = [-1, 150, 160, 170, -1, -1, 180, 190]$.

2. Project 8 - Students

Write a program to:

- Input the number of students in class.
- Input the name of students in class, mark.
- Sort students due to their mark.

1.2 Đề tài làm thêm

1. Project 21 - Digit Degree

Let's define the digit degree of some positive integer as the number of times we need to replace this number with the sum of its digits until we get to a one digit number. Given an integer, find its digit degree.

Example:

- For $n = 5$, the output should be $\text{digitDegree}(n) = 0$
- For $n = 100$, the output should be $\text{digitDegree}(n) = 1$
- For $n = 91$, the output should be $\text{digitDegree}(n) = 2$

2. Project 23 - Surpassing Words

Surpassing words are English words for which the gap between each adjacent pair of letters strictly increases. These gaps are computed without "wrapping around" from Z to A.

For example:

Write a function to determine whether a word passed into a function is a surpassing word. You can assume the word is made of only alphabetic characters, and are separated by whitespace. We will consider the empty string and a 1-character string to be a valid surpassing word.

- $\text{is_surpassing_word}(\text{"superb"}) \Rightarrow \text{True}$
- $\text{is_surpassing_word}(\text{"excellent"}) \Rightarrow \text{False}$

3. Project 24 - Cyclone Word (Challenge)

Cyclone Word (challenge) Cyclone words are English words that have a sequence of characters in alphabetical order when following a cyclic pattern.

For example:

Write a function to determine whether a word passed into a function is a cyclone word. You can assume that the word is made of only alphabetic characters, and is separated by whitespace.

- `is_cyclone_phrase("adjourned") => True`

- `is_cyclone_phrase("settled") => False`

1.3 Công cụ sử dụng

Mars4_5

1.4 Source Code

<https://github.com/thanhff/Computer-Architecture-Lab/tree/master/Mini-project>

2 Project 7 – Sort By Height

Đề bài: Some people are standing in a row in a park. There are trees between them which cannot be moved. Your task is to rearrange the people by their heights in a nondescending order without moving the trees. People can be very tall!

Example:

For $a = [-1, 150, 190, 170, -1, -1, 160, 180]$, the output should be $\text{sortByHeight}(a) = [-1, 150, 160, 170, -1, -1, 180, 190]$.

2.1 Phân tích cách thức thực hiện

Phân tích đề bài:

- Đầu vào: mảng array chứa các giá trị.
- Chú ý: giá trị -1 biểu thị cho tree (cây).
- Đầu ra: mảng các giá trị được sắp xếp tăng dần (các giá trị -1 giữ nguyên vị trí của nó).

Ý tưởng:

- **Bước 1:** @find_height: Lấy các giá trị là chiều cao của con người cho vào mảng height.
- **Bước 2:** @sort_height: Sắp xếp mảng height theo thứ tự tăng dần (theo thuật toán Bubble Sort).
- **Bước 3:** @change: Thay thế những giá trị đã được sắp xếp trong mảng height vào mảng array.

2.2 Ý nghĩa của các thanh ghi được sử dụng

- \$a0: lưu địa chỉ của array
- \$a1: lưu địa chỉ height
- \$s0: lưu giá trị -1 (biểu thị cho tree)
- \$s1: n - số lượng phần tử của array
- \$s2: m - số lượng phần tử của height

1. Trong @find_height:

- \$t0: biến i
- \$t2: biến j

2. Trong @sort_height:

- \$t0: biến i
- \$t1: biến j
- \$t2: m - i - 1

3. Trong @change:

- \$t0: biến i
- \$t2: biến j

2.3 Source code

```
1 #-----
2 # Project 7: sortByHeight
3 # @input: array (mang chua cac gia tri dau vao)
4 # @note: cac gia tri -1 bieu thi tree
5 # Mang height: chi chua cac chieu cao lay tu mang array
6 # @idea: Input array
7 # 1. @find_height: Lay cac gia tri la chieu cao cua nguoi => height
8 # 2. @sort_height: Sap xep mang height theo thu tu tang dan
9 # 3. @change: Thay the nhung gia tri da duoc sap xep trong mang height
   vao mang array
10 #-----
11
12 .data
13 # Input array
14 # @note: Chu y so luong phan tu 'n' cua mang trong $s1
15 array: .word -1, 150, 190, 170, -1, -1, 160, 180
16
17 # Luu cac gia tri height cua nguoi ra mot mang moi
18 height: .word
19
20
21 .text
22 main:
23     la      $a0, array      # Lay dia chi cua array cho vao $a0
24     la      $a1, height     # Lay dia chi height cho vao $a1
25     addi    $s0, $zero, -1   # -1: bieu thi Tree trong array input
26     addi    $s1, $zero, 8    # n: So luong phan tu cua array
27     addi    $s2, $zero, 0    # m: So luong phan tu cua height
28
29     j find_height
30
31 after_find_height:
32     j sort_height
33
34 after_sort:
35     j change
36
37 after_change:
38     li $v0, 10
39     syscall
40 end_main:
41
42
43 #-----
```

```

44 # 1. @find_height: Lay cac gia tri la chieu cao cua nguoi => height
45 # @input: array (mang chua gia tri dau vao)
46 # @output: mang height chua gia tri chieu cao cua nguoi
47 #-----
48
49 find_height:
50
51 # Khoi tao cac bien i, j bang 0
52 addi $t0, $zero, 0 # i = 0
53 addi $t2, $zero, 0 # j = 0
54
55 fh_loop:
56 sll $t1, $t0, 2 # $t1 = 4*i
57 add $t1, $t1, $a0 # Vi tri cua input[i]
58 lw $s3, 0($t1) # Lay gia tri cua input[i]
59
60 beq $s3, $s0, fh_continue # Neu gia tri input[i] == -1 => continue
    (bo qua tree)
61
62 sll $t3, $t2, 2 # $t3 = 4*j
63 add $t3, $t3, $a1 # Vi tri cua height[j]
64 sw $s3, 0($t3) # Luu gia tri vao vi tri height[j]
65 addi $t2, $t2, 1 # j = j + 1
66 addi $s2, $s2, 1 # m = m + 1 (so luong phan tu trong height tang
    len 1)
67
68 fh_continue:
69 addi $t0, $t0, 1 # i = i + 1
70 slt $t4, $t0, $s1 # if i < n => True: return 1; False: return 0
71 bne $t4, $zero, fh_loop
72
73 fh_end_loop:
74 j after_find_height
75
76 ##### Sap xep cac phan tu trong mang height
77 ### BubbleSort
78
79 #-----
80 # 2. @sort_height: Sap xep mang height theo thu tu tang dan
81 # @case_study: Sap xep bang phuong phap 'BubbleSort'
82 # @input: height (chua gia tri chieu cao cua nguoi) - chua duoc sap
    xep
83 # @output: height (chua gia tri chieu cao cua nguoi) - da sap xep theo
    thu tu tang dan
84 #-----
85
86 sort_height:
87
88 # Khoi tao index i cua loop_1 bang 0
89 addi $t0, $zero, 0 # i = 0
90
91 loop_1:

```

```

92  # Khoi tao index j cua loop_2 bang 0
93  addi    $t1, $zero, 0    # j = 0
94
95  addi    $t0, $t0, 1      # i = i + 1
96  sub     $t2, $s2, $t0    # m - i - 1
97
98  # Kiem tra dieu kien: i < m - 1
99  slt     $t6, $t0, $s2
100 beq     $t6, $zero, end_loop_1
101
102 loop_2:
103  # Kiem tra dieu kien j < m - i - 1
104  slt     $t5, $t1, $t2    # j < m - i - 1: True return 1; else return 0
105  beq     $t5, $zero, end_loop_2
106
107  sll     $t3, $t1, 2      # $t3 = 4*j
108  add     $t3, $t3, $a1    # Vi tri cua A[j]
109  lw      $s3, 0($t3)      # Lay gia tri cua A[j]
110  lw      $s4, 4($t3)      # Lay gia tri cua A[j+1]
111
112  if:
113  slt     $t4, $s3, $s4    # Kiem tra A[j] < A[j+1] => True: return
114      1; False: return 0
115  bne     $t4, $zero, end_if
116
117  # Neu A[j] > A[j + 1] => Thuc hien Swap 2 phan tu nay
118  sw      $s4, 0($t3)
119  sw      $s3, 4($t3)
120
121  end_if:
122  addi    $t1, $t1, 1      # j = j + 1
123  j       loop_2
124
125  end_loop_2:
126  j       loop_1
127
128  end_loop_1:
129  j       after_sort
130
131  #-----
132  # 3. @change: Thay the nhung gia tri da duoc sap xep trong mang height
133      vao mang array
134  # @input: array (mang chua gia tri dau vao)
135  # @output: array (mang chua gia tri duoc sap xep theo chieu cao cua
136      nguoi)
137  # @note: Reset cac gia tri cua height = 0
138  #-----
139
140  change:
141  # Khoi tao bien i, j bang 0
142  addi    $t0, $zero, 0    # i = 0

```



```

141     addi    $t2, $zero, 0    # j = 0
142
143 i_loop:
144     sll     $t1, $t0, 2      # $t1 = 4*i
145     add     $t1, $t1, $a0    # Vi tri cua array[i]
146     lw      $s3, 0($t1)      # Lay gia tri cua array[i]
147
148     beq     $s3, $s0, i_continue # Neu gia tri array[i] == -1 => continue
149
150     sll     $t3, $t2, 2      # $t3 = 4*j
151     add     $t3, $t3, $a1    # Vi tri cua height[j]
152     lw      $s3, 0($t3)      # Lay gia tri cua height[j]
153     sw      $s3, 0($t1)      # array[i] = height[j]
154
155     addi    $s4, $zero, 0
156     sw      $s4, 0($t3)      # Reset gia tri cua height[j] = 0
157     addi    $t2, $t2, 1      # j = j + 1
158
159 i_continue:
160     addi    $t0, $t0, 1      # i = i + 1
161     slt     $t4, $t0, $s1    # if i < n: True return: 1; False return: 0
162     bne     $t4, $zero, i_loop
163
164 i_end_loop:
165     j       after_change
166
167 #-----
168 # END
169 #-----

```

Project 07 - Sort By Height

2.4 Kết quả chạy chương trình

- Đầu vào:

```
1 array: .word -1, 150, 190, 170, -1, -1, 160, 180
```

Data Segment								
Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	-1	150	190	170	-1	-1	160	180
0x10010020	0	0	0	0	0	0	0	0
0x10010040	0	0	0	0	0	0	0	0
0x10010060	0	0	0	0	0	0	0	0
0x10010080	0	0	0	0	0	0	0	0

- Kết quả:

Data Segment								
Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	-1	150	160	170	-1	-1	180	190
0x10010020	0	0	0	0	0	0	0	0
0x10010040	0	0	0	0	0	0	0	0
0x10010060	0	0	0	0	0	0	0	0
0x10010080	0	0	0	0	0	0	0	0
0x100100a0	0	0	0	0	0	0	0	0

3 Project 8 – Students

Đề bài: Write a program to:

- Input the number of students in class.
- Input the name of students in class, mark .
- Sort students due to their mark.

3.1 Phân tích cách thức thực hiện

Phân tích đề bài:

- Input: số sinh viên và danh sách sinh viên.
- Output: Danh sách sinh viên được sắp xếp theo điểm (tăng dần).

Ý tưởng:

- Nhập số lượng sinh viên (số nguyên dương)
- Nhập lần lượt tên và điểm sinh viên với điều kiện: $0 \leq \text{điểm} \leq 10$
- Sắp xếp danh sách sinh viên theo thứ tự tăng dần của điểm sử dụng thuật toán Bubble sort.

Cách thức thực hiện:

- **Bước 1:** Nhập n (nếu n không phải số nguyên dương thì nhập lại)
- **Bước 2:** Nhập tên và điểm sinh viên (nếu điểm nhập vào không thỏa mãn: $0 \leq \text{điểm} \leq 10$ thì nhập lại)
- **Bước 3:** Hiển thị thông tin sinh viên vừa nhập vào.
- **Bước 4:** Sắp xếp điểm sinh viên:
 - + $j = 0$ //xử lý lần đầu tiên
 - + $i = n$ //duyệt từ cuối dãy trở về vị trí
 - Trong khi $i > j$ thực hiện:
 - Nếu $a[i] < a[i-1]$ thì hoán đổi $a[i]$ và $a[i-1]$
 - $i = i - 1$
 - + Nếu $j = j + 1$
 - Nếu $j > n - 1$ thì dừng
 - Ngược lại lặp lại bước 2.
- **Bước 5:** Hiển thị danh sách sinh viên đã được sắp xếp.

3.2 Ý nghĩa của các thanh ghi được sử dụng

- \$s1: lưu địa chỉ của mảng student
- \$t1: gán địa chỉ của mảng vào \$t1
- \$a0: số sinh viên
- \$s0, \$s6, \$s7: số sinh viên
- \$a1: chỉ ra vị trí lưu tên

- \$f1: đọc giá trị của \$s4 (=10.0) vào \$f1
- \$t0: biến đếm của sinh viên được nhập thông tin
- \$t1: chứa giá trị của mảng được gán
- \$s3: lưu địa chỉ của students để thực hiện sắp xếp
- \$t2: biến i
- \$t3: biến j
- \$t4: j - 1

3.3 Ý nghĩa của chương trình con

- __read_info_student: nạp địa chỉ của mảng.
- count: nhập số lượng sinh viên.
- name: nhập tên sinh viên với giới hạn là 47 kí tự.
- mark: nhập điểm sinh viên với điều kiện $0 \leq \text{điểm} \leq 10$.
- __bubble_sort: thực hiện sắp xếp.
- __show_student: Hiển thị danh sách sau khi sắp xếp.

3.4 Source code

```

1 #-----
2 # Project 8: Students
3 #@idea:
4 # - Nhập số lượng sinh viên.
5 # - Nhập tên sinh viên.
6 # - Nhập điểm sinh viên với điều kiện 0 <= điểm <= 10.
7 # - Sắp xếp nhân viên theo điểm dùng thuật toán bubble sort.
8 #-----
9
10 .data
11 students: .space 5200          # Array to store blocks (52) of 100
12         student
13 input_name: .asciiz "Nhập tên sinh viên"
14 input_mark: .asciiz "Nhập điểm sinh viên"
15 note: .asciiz "Điểm được nhập phải thỏa mãn: 0 <= điểm <= 10"
16 input_count: .asciiz "Nhập số lượng sinh viên"
17 error_count: .asciiz "Số sinh viên phải là số tự nhiên nhỏ hơn hoặc
18         bang 100"
19 error: .asciiz "Giá trị nhập vào không đúng kiểu!"
20 error_in: .asciiz "Tên bạn đã nhập quá dài hoặc không nhập gì! Tôi
21         đã 47 kí tự"
22 st_list: .asciiz "Danh sách sinh viên"
23 mark_and_name: .asciiz "\nĐiểm\tHọ và Tên\n"
24 sorted_list: .asciiz "\nDanh sách đã sắp xếp:\n"
25 so_sinh_vien: .word
26 ten: .float 10.0
27 zero: .float 0.0

```

```

26 .text
27 #Doc thong tin sinh vien
28 __read_info_student:
29     la $s1, students          # Nap dia chi cua mang $s1
30     move $t1, $s1             # Gan dia chi cua mang vao $t1
31     la $s4, ten               # $s4 = 10.0
32     la $s5, zero              # $s5 = 0.0
33     li $t2, 101               # $t2 = 101
34
35 count:
36     li $v0, 51                # Goi hop thoai nhap so luong sinh vien
37     la $a0, input_count       # $a0 tieu de: "Nhap so luong sinh vien"
38     syscall                   # $a0 so sinh vien
39
40     addi $t0, $zero, -1
41     beq $t0, $a1, error_input
42     slt $t0, $a0, $zero        # neu gia tri nhap vao < 0 => return 1
43     bne $t0, $zero, error_input
44
45     slt $v0, $a0, $t2          # Neu a0 (so sinh vien) < t2 ( = 100 ) --> v0
                                # = 1
46     beqz $v0, nhap_lai        # Neu v0 == 0 --> Nhap lai
47
48     j end_error_input
49
50 nhap_lai:
51     li $v0, 55                # Goi hop thoai Thong bao
52     la $a0, error_count
53     li $a1, 0                 # Hop thoai Error
54     syscall
55     j count                   # Quay lai ham Nhap so SV
56 error_input:
57     la $v0, 55
58     la $a0, error
59     syscall
60     j count
61 #Thoat khoi ham nhap so sinh vien
62 end_error_input:
63     move $s0, $a0             # ao = so sinh vien
64     move $s7, $a0             # s0,s6,s7 = so sinh vien
65     move $s6, $a0
66
67     li $v0, 4
68     la $a0, st_list           # In ra chuoi "Danh sach sinh vien "
69     syscall                   # tren giao dien console
70
71     la $a0, mark_and_name     # in ra chuoi "Diem Ho va Ten"
72     syscall
73
74     li $t0, 0                 # khoi tao $t0 = 0, $t0 la bien dem (i) cua sinh
                                # vien vua duoc nhap thong tin
75 loop:

```

```

76  slt    $v0, $t0, $s0      # So sanh $t0 (So sinh vien nhap thong tin
    ) < $s0 (Tong so sinh vien)-> v0 = 1
77  beqz   $v0, end_loop     # Thoat vong lap khi nhap du thong tin cho
    cac sinh vien
78
79  ### Nhap ten sinh vien
80  name:
81  li     $v0, 54            # Goi hop thoai nhap ten sinh vien
82  la     $a0, input_name    # Tieu de "Nhap ten sinh vien"
83  la     $a1, 4($t1)        # Chi ra vi tri luu ten
84  li     $a2, 46            # Gioi han do dai ten 46 ki tu
85  syscall
86  bnez   $a1, re_input      # Neu a1 != 0 --> Nhap ten qua dai ( a1 =
    0 la trang thai dung ) --> Nhap lai name
87  j      mark              # Chay den ham nhap diem
88
89  ### Nhap lai ten
90  re_input:
91  li     $v0, 55            # Goi hop thoai thong bao String
92  la     $a0, error_input   # Goi string loi
93  li     $a1, 0             # Goi hop thoai error
94  syscall
95  j      name              # Quay lai Nhap name
96  mark:
97
98  do: li  $v0, 52            # Goi hop thoai nhap diem (Kieu float)
99  la     $a0, input_mark    # Tieu de "Nhap diem sinh vien"
100  syscall
101
102
103  bnez   $a1, error_inpu
104
105  l.s    $f1,($s4)          # Doc gia tri cua s4 (= 10.0) vao thanh ghi f1
106  c.le.s $f0, $f1          # f0 <= f1
107  li     $a0, 1
108  movt   $a0, $zero
109  bne    $a0, $zero, condi  # Neu a0 != 0 --> incre
110
111  l.s    $f1,($s5)
112  c.le.s $f1, $f0          # f1 <= f0
113  li     $a0, 1
114  movt   $a0, $zero
115  beq    $a0, $zero, exit_do # Neu a0 != 0 --> incre
116  error_inpu:
117  la     $v0, 55
118  la     $a0, error
119  syscall
120  j      do
121  condi:
122
123  la     $v0, 55
124  la     $a0, note

```

```

125 syscall
126 j    do
127
128 exit_do:
129 s.s   $f0, ($t1)      # Luu diem vao mang
130 li    $v0, 2          #Goi ham in ra man hinh kieu float
131 mov.s  $f12, $f0      # In diem ra man hinh console
132 syscall
133 li    $v0, 11         #in ky tu
134 li    $a0, '\t'       # In dau tab
135 syscall
136 li    $v0, 4          #in string
137 la    $a0, 4($t1)     # In ten sinh vien ra man hinh console
138 syscall
139 addi   $t0, $t0, 1     #Tang bien dem len 1
140 addi   $t1, $t1, 52    # Chuyen sang vung nho tiep theo, moi vung
141      nho 52 bit
142 j    loop             # Lap lai vong lap
143 end_loop:
144
145 #-----
146 # Bat dau sap xep #
147 #@case_study: Sap xep bang phuong phap bubble sort
148 #@input: danh sach sinh vien - chua duoc sap xep theo diem
149 #@output: danh sach sinh vien - da duoc sap xep theo thu tu tang dan
150      cua diem so
151 #-----
152
153 __bubble_sort:
154 la    $s3, students   #load dia chi mang a vao $s3
155 addi   $t2, $t0, 0     #i = n
156 loop1: # for i = n-1 to 0
157 addi   $t2, $t2, -1    # i = i - 1
158 add    $s0, $s3, $zero #Cho dia chi cua mang s0 = s3
159 li    $t3, 0          #gan j = 0
160 beq    $t2, 0, break_1 # so sanh i voi 0 neu bang nhau thi re
161      nhanh xuong nhan break
162 loop2: #for j = 0 to i - 1
163 beq    $t2, $t3, loop1 # if j == i then loop1
164 l.s    $f1, 0($s0)     # Load a[j] luu vao $f1
165 addi   $s0, $s0, 52    # tang dia chi len 52 --> Chuyen sang
166      student khac --> s0 = a[j+1] = f2
167 l.s    $f2, 0($s0)     # Load a[j+1] = f2
168 c.lt.s $f2, $f1       # Neu f1 < f2 @@@
169 li    $a0 , 1
170 movt   $a0 , $zero
171 bne    $a0 , $zero ,   incre      # Neu f2 >= f1 --> incre
172 jal    swap           # Neu f2 < f1 --> swap
173 incre:
174 addi   $t3, $t3, 1     # j = j + 1
175 j    loop2            #nhay vao loop2
176

```

```

173 break_1:
174
175     j    __show_student    #Show student
176     ### a[j-1] vs a[j]
177     swap:
178         li    $s7, 0
179         addi   $t4, $s0, -52    # t4 = students[j-1] --> f1 --> Vung nho dau
                                   tien
180     loopx:
181         slti   $v0, $s7, 13    #s7 < 13 -->v0 = 1
182         beqz   $v0, end_loopx    #v0 = 0 -> s7 >= 13 -> Thoat vong lap
183         lw     $a1, 0($t4)      # Load gia tri cua f1 ( a[j-1] ) vao a1
184         lw     $a2, 52($t4)     # Load gia tri cua f2 ( a[j] ) vao a2
185         sw     $a1, 52($t4)     # Ghi gia tri cua a1 vao a[j]
186         sw     $a2, 0($t4)      # Ghi gia tri cua a2 vao a[j-1]
187         addi   $t4, $t4, 4      # Tang vung nho cua students[j-1]
188         addi   $s7, $s7, 1      # Tang bien dem len 1
189         j     loopx
190     end_loopx:
191         jr     $ra              #quay ve incre
192
193     ### in ra danh sach sinh vien da duoc sap xep
194     __show_student:
195         add     $s0, $s7, $zero    #so sinh vien = 0
196         la      $t1, students      # Nap dia chi dau mang cac block luu thong
                                   tin sinh vien
197         li      $v0, 4             # Goi ham in string
198         la      $a0, sorted_list
199         syscall
200         la      $a0, mark_and_name    # in ra chuoi "Diem Ho va Ten"
201         syscall
202         li      $t0, 0             # Khoi tao $t0 = 0, $t0 la bien diem so sinh vien
                                   da duoc duyet
203     loop3:
204         slt     $v0, $t0, $s6      # So sanh $t0 (So sv da duyet) va $s0 (
                                   Tong so sinh vien)
205         beqz   $v0, exit          # Thoat vong lap khi duyet het sinh vien
206
207         li      $v0, 2
208         l.s     $f12, 0($t1)       # In diem ra man hinh console
209         syscall
210         li      $v0, 11           # in char
211         li      $a0, '\t'         # In dau tab
212         syscall
213         li      $v0, 4            # in string
214         la      $a0, 4($t1)       # In ten sinh vien ra man hinh console
215         syscall
216     continue:
217         addi   $t0, $t0, 1        # Tang bien dem so luong sv da duyet
218         addi   $t1, $t1, 52       #tang dia chi t1 len 52
219         j     loop3              # Lap lai vong lap
220     exit:

```

```

221 li $v0, 10          #Thoat chuong trinh
222 syscall
223 #-----
224 # END
225 #-----

```

Project 08 - Students

3.5 Kết quả chạy chương trình

- Ban đầu:

Danh sach sinh vien	
Diem	Ho va Ten
9.0	Nguyen Kim Anh
5.5	Nguyen Quang Huy

- Sau khi sắp xếp:

Danh sach da sap xep:	
Diem	Ho va Ten
5.5	Nguyen Quang Huy
9.0	Nguyen Kim Anh

4 Project 21 – Digit Degree

Đề bài: Project 21 - Digit Degree

Let's define the digit degree of some positive integer as the number of times we need to replace this number with the sum of its digits until we get to a one digit number. Given an integer, find its digit degree.

Example:

- For $n = 5$, the output should be $\text{digitDegree}(n) = 0$
- For $n = 100$, the output should be $\text{digitDegree}(n) = 1$
- For $n = 91$, the output should be $\text{digitDegree}(n) = 2$

4.1 Phân tích cách thức thực hiện

Phân tích đề bài:

- Đầu vào: một số nguyên dương.
- Ban đầu: $\text{digit_degree} = 0$.
- Đầu ra: digit degree của số nguyên nhập vào đó.

Cách thức thực hiện:

Bước 1: Check: nếu số $< 10 \Rightarrow$ return digit_degree . Nếu $> 10 \Rightarrow \text{digit_degree} += 1$ và tiếp tục bước 2.

Bước 2: Sử dụng vòng lặp để tách các chữ số bằng cách chia cho 10 và lưu vào mảng number.

Bước 3: Count: tính tổng các chữ số quay lại Bước 1 check để kiểm tra.

4.2 Ý nghĩa của các thanh ghi được sử dụng

- \$a1: giá trị của số được nhập vào
- \$a0: vị trí của mảng number
 - Trong digit_degree
 - \$v0: đầu ra của chương trình con
 - \$s0: lưu giá trị của 10 để so sánh
 - \$s1: giá trị đầu vào của chương trình
 - \$s2: tổng của các chữ số sau khi số đó được tách

4.3 Ý nghĩa của chương trình con

- digit_degree : thực hiện tính toán số lượng digit degree của số nhập vào.

4.4 Source code

```

1 #-----
2 # Project 21 - Digit Degree
3 # @input: mot so nguyen duong
4 # digit_degree ban dau bang 0
5 # @idea:
6 # 1. Check: Neu so < 10: return digit_degree
7 # 2. Su dung vong lap de tach cac chu so bang cach chia dan cho 10 va
   luu vao mang number.
8 # 3. Count: tinh tong cac chu so, neu tong > 10: digit_degree cong
   them 1
9 # quay lai 'check' de kiem tra
10 #-----
11
12
13 .data
14 notice: .asciiz "Nhap gia tri so nguyen: "
15 result: .asciiz "digitDegree = "
16 error: .asciiz "Gia tri nhap vao khong dung kieu!"
17 number: .word
18
19 .text
20 main:
21 # Nhap gia tri so nguyen
22 li $v0, 51
23 la $a0, notice
24 syscall
25
26 # Kiem tra input nhap vao co phai la so nguyen duong hay khong
27 addi $t0, $zero, -1
28 beq $t0, $a1, error_input
29 slt $t0, $a0, $zero # neu gia tri nhap vao < 0 => return 1
30 bne $t0, $zero, error_input
31
32 j end_error_input
33 error_input:
34 la $v0, 55
35 la $a0, error
36 syscall
37 j end_main
38
39 end_error_input:
40
41 add $a1, $a0, $zero # So nhap vao duoc luu vao $a1
42 la $a0, number # Luu vi tri cua number
43 jal digit_degree
44 nop
45
46 # Hien thi ket qua
47 add $a1, $v0, $zero
48 li $v0, 56
49 la $a0, result
50 syscall

```

```

51
52 end_main:
53     li $v0, 10
54     syscall
55
56 #-----
57 # Thuc hien tinh digit_degree
58 #-----
59 digit_degree:
60     addi $v0, $zero, 0    # Dau ra cua chuong trinh con
61     addi $s0, $zero, 10   # $s0 luu gia tri 10 de so sanh
62     add $s1, $zero, $a1   # $s1 gia tri dau vao cua chuong trinh
63
64 #-----
65 # 1. Check: Neu so < 10: return digit_degree
66 #-----
67
68 check:
69     slt $t0, $s1, $s0     # if number < 10 => return 1
70     bne $t0, $zero, end_digit_degree
71     addi $v0, $v0, 1      # Neu so do > 10 => degit_degree += 1
72
73 #-----
74 # 2. Su dung vong lap de tach cac chu so bang cach chia dan cho 10 va
75 #    luu vao mang number.
76 #-----
77     addi $t7, $zero, 0    # So luong cac chu so n
78 loop:
79     div $t1, $s1, 10
80     mfhi $t1
81
82     add $t2, $t7, $a0     # vi tri cua phan tu
83     sb $t1, 0($t2)
84     addi $t7, $t7, 1
85
86     div $s1, $s1, 10
87     beq $s1, $zero, end_loop
88
89     j loop
90
91 end_loop:
92
93 #-----
94 # 3. Count: tinh tong cac chu so, neu tong > 10: digit_degree cong
95 #    them 1
96 #    quay lai 'check' de kiem tra
97 #-----
98     addi $s2, $zero, 0    # tong cua cac chu so
99     addi $t3, $zero, 0    # i = 0
100 for:

```

```

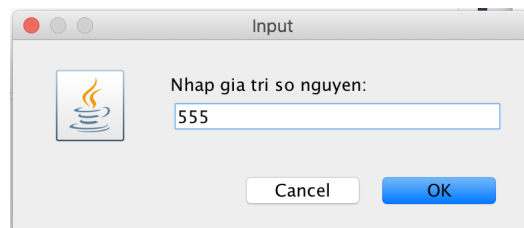
101 add $t5, $t3, $a0    # vi tri
102 lb $s3, 0($t5)
103
104 add $s2, $s2, $s3    # sum = sum + a[i]
105
106 addi $t6, $zero, 0    # xoa bo
107 sb $t6, 0($t5)
108
109 addi $t3, $t3, 1      # i = i + 1
110 slt $t4, $t3, $t7     # i >= n => False 0
111 beq $t4, $zero, end_for
112 j for
113
114 end_for:
115 sle $t0, $s0, $s2     # if 10 <= $s2 => return 1
116 beq $t0, $zero, end_digit_degree
117
118 add $s1, $zero, $s2    # tong moi sau khi cong cac chu so
119
120 j check
121
122 end_digit_degree:
123 jr $ra
124
125 #-----
126 # END
127 #-----

```

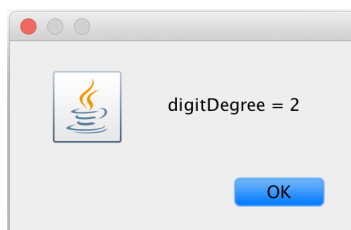
Project 21 - Digit Degree

4.5 Kết quả chạy chương trình

- Đầu vào:



- Kết quả:



5 Project 23 – Surpassing Words

Đề bài: Project 23 - Surpassing Words

Surpassing words are English words for which the gap between each adjacent pair of letters strictly increases. These gaps are computed without "wrapping around" from Z to A.

For example:

Write a function to determine whether a word passed into a function is a surpassing word. You can assume the word is made of only alphabetic characters, and are separated by whitespace. We will consider the empty string and a 1-character string to be a valid surpassing word.

- `is_surpassing_word("superb") => True`
- `is_surpassing_word("excellent") => False`

5.1 Phân tích cách thức thực hiện

Phân tích đề bài:

- Đầu vào: một từ tiếng Anh bất kỳ.
- Đầu ra: từ đó có phải surpassing words hay không.

Cách thức thực hiện:

- **Bước 1:** Sử dụng vòng lặp for từ $i = 0$ cho đến $i = n - 2$ để so sánh 2 ký tự đứng liền nhau. Mỗi lần so sánh 2 ký tự liền nhau này \Rightarrow lưu kết quả vào mảng `gap_length`. (Trong trường hợp số âm \Rightarrow Chuyển sang số dương).
- **Bước 2:** Duyệt mảng `gap_length` để kiểm tra. Nếu gặp bất kỳ trường hợp nào mà `gap_length[i] > gap_length[i+1]` \Rightarrow Trả về kết quả False. Duyệt hết mảng mà không gặp trường hợp nào như trên thì trả về True.

5.2 Ý nghĩa của các thanh ghi được sử dụng

- `$a0`: số ký tự của `input_word`
- `$a1`: lưu vị trí của `input_word`
- `$a2`: lưu vị trí của `gap_length`
- Trong vòng lặp loop thứ 1:
 - `$t0`: i
 - `$t1`: $n - 1$
- Trong vòng lặp loop thứ 2:
 - `$t0`: i
 - `$t1`: $n - 2$

5.3 Ý nghĩa của chương trình con

- check: kiểm tra từ đó có tuân thủ theo quy tắc của Surpassing Words hay không.

5.4 Source code

```
1 #-----
2 # Project 23: Surpassing Words
3 # @input: mot tu bat ky luu vao input_word
4 # @idea:
5 # Buoc 1: Su dung 1 vong lap for tu i = 0 cho den i = n - 2 de so sanh
   2 ky tu dung lien nhau
6 #   Moi lan so sanh 2 ki tu lien nhau => Luu vao mang gap_length
7 #   Trong truong hop so am => Dao dau sang so duong
8 # Buoc 2: Duyet mang gap_length kiem tra
9 #   Neu gap bat ki truong hop nao ma gap_length[i] > gap_length[i+1]
   => FALSE
10 #   Duyet het mang ma khong gap truong hop tren => TRUE
11 #-----
12
13 .data
14   # Chu y so luong phan tu cua tu
15   input_word: .asciiz "superb"
16   true: .asciiz "True"
17   false: .asciiz "False"
18   gap_length: .word
19
20 .text
21 main:
22   addi $a0, $zero, 6   # n: So cac ki tu cua input_word
23   la $a1, input_word
24   la $a2, gap_length
25
26
27   jal check
28   nop
29
30   beq $v0, $zero, false_answer
31
32 true_answer:
33   li $v0, 55
34   la $a0, true
35   syscall
36   j end_main
37
38 false_answer:
39   li $v0, 55
40   la $a0, false
41   syscall
42
```

```

43 end_main:
44     li $v0, 10
45     syscall
46
47
48 #-----
49 # @check: Kiem tra xem co phai Surpassing words hay khong
50 #-----
51
52
53 #-----
54 # Buoc 1: Su dung 1 vong lap for tu i = 0 cho den i = n - 2 de so sanh
55 #       2 ky tu dung lien nhau
56 #       Moi lan so sanh 2 ki tu lien nhau => Luu vao mang gap_length
57 #       Trong truong hop so am => Dao dau sang so duong
58 #-----
59 check:
60     addi $t0, $zero, 0 # i = 0
61     sub $t1, $a0, 1    # $t1 = n - 1
62 loop:
63
64     addi $t4, $t0, 1 # j = i + 1
65
66     add $s0, $a1, $t0
67     lb $s0, 0($s0)   # A[i]
68
69     add $s1, $a1, $t4
70     lb $s1, 0($s1)   # A[i+1]
71
72     sub $s2, $s0, $s1
73
74     slt $t5, $s2, $zero # if $s2 > 0 => 0
75     beq $t5, $zero, continue
76
77     nor $s2, $s2, $zero # Neu $s2 < 0 => Dao dau $s2 => Duong
78     add $s2, $s2, 1
79
80
81 continue:
82     add $t2, $t0, $a2
83     sb $s2, 0($t2)
84
85     addi $t0, $t0, 1 # i = i + 1
86     slt $t3, $t0, $t1 # i < n - 1 => 1; false => 0
87     beq $t3, $zero, end_loop
88     j loop
89
90 end_loop:
91
92 #-----
93 # Buoc 2: Duyet mang gap_length kiem tra

```

```

94 # Neu gap bat ki truong hop nao ma gap_length[i] > gap_length[i+1]
    => FALSE
95 # Duyet het mang ma khong gap truong hop tren => TRUE
96 #-----
97
98     addi $t0, $zero, 0    # i = 0
99     sub $t1, $a0, 2      # $t1 = n - 2
100
101 loop_2:
102     addi $t4, $t0, 1     # i + 1
103
104     add $s0, $a2, $t0
105     lb $s0, 0($s0)      # gap_length[i]
106
107     add $s1, $a2, $t4
108     lb $s1, 0($s1)      # gap_length[i+1]
109
110     slt $t2, $s0, $s1    # gap_length[i] < gap_length[i+1] => 1
111     beq $t2, $zero, return_false
112
113     addi $t0, $t0, 1     # i = i + 1
114     slt $t3, $t0, $t1    # i < n - 2 => 1; false => 0
115     beq $t3, $zero, end_loop_2
116
117     j     loop_2
118 end_loop_2:
119
120 return_true:
121     addi $v0, $zero, 1
122     jr $ra
123 return_false:
124     addi $v0, $zero, 0
125     jr $ra
126
127 #-----
128 # END
129 #-----

```

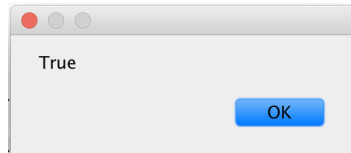
Project 23 - Surpassing Words

5.5 Kết quả chạy chương trình

- Đầu vào:

```
1 input_word: .asciiz "superb"
```

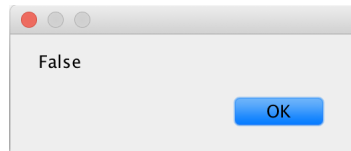
- Kết quả:



- Đầu vào:

```
1 input_word: .asciiz "excellent"
```

- Kết quả:



6 Project 24 – Cyclone Word (Challenge)

Đề bài: Project 24 - Cyclone Word (Challenge)

Cyclone Word (challenge) Cyclone words are English words that have a sequence of characters in alphabetical order when following a cyclic pattern.

For example:

Write a function to determine whether a word passed into a function is a cyclone word. You can assume that the word is made of only alphabetic characters, and is separated by whitespace.

- `is_cyclone_phrase("adjourned") => True`
- `is_cyclone_phrase("settled") => False`

6.1 Phân tích cách thức thực hiện

Phân tích đề bài:

- Đầu vào: một từ tiếng Anh bất kỳ.
- Đầu ra: từ đó có phải Cyclone Word hay không.

Cách thức thực hiện:

- **Trường hợp 1:** Nếu từ nhập vào có độ dài bằng 0 hoặc bằng 1 => TRUE.
- **Trường hợp 2:** Nếu từ nhập vào có độ dài > 1:
 - + Sử dụng vòng lặp có $i = 0$ (vị trí đầu tiên) và $j = n - 1$ (vị trí cuối cùng của từ đó).
 - + Thực hiện so sánh `c_word[i]` với `c_word[j]` và `c_word[i+1]` với `c_word[j]` và cứ tiếp tục cho đến khi $i \geq j$. Nếu `c_word[i] <= c_word[j]` và `c_word[j] <= c_word[i+1]` => TRUE.

6.2 Ý nghĩa của các thanh ghi được sử dụng

- `$a0`: lưu vị trí của `c_word`
- `$a1`: số lượng kí tự của từ đó
 - Trong `cyclone_word`:
 - `$v0`: kết quả trả về của chương trình
 - `$t0`: i (ban đầu bằng 0)
 - `$t1`: $j = n - 1$

6.3 Ý nghĩa của chương trình con

- `cyclone_word`: kiểm tra xem từ đó có phải Cyclone word hay không.

6.4 Source code

```
1 #-----
2 # Project 24: Cyclone Word (Challenge)
3 # @input: mot tu bat ky luu vao c_word
4 # @idea:
5 #   TH1: Neu tu nhap vao co do dai bang 0 hoac bang 1 => TRUE
6 # TH2: Neu tu nhap vao co do dai > 1:
7 #       + Su dung vong lap co i bat dau bang 0 (Vi tri dau tien)
8 #       + va j = n - 1 (vi tri cuoi cung cua tu do)
9 #       + Thuc hien so sanh c_word[i] voi c_word[j] va c_word[i+1] voi
10 #        c_word[j] va cu tiep tục cho den khi i >= j
11 #        Neu c_word[i] <= c_word[j] va c_word[j] <= c_word[i+1] =>
12 #        TRUE
13 #-----
14 .data
15 # Chu y so luong ki tu cua tu
16 c_word: .asciiz "adjourned" # input dau vao
17 true: .asciiz "True"
18 false: .asciiz "False"
19 .text
20
21 main:
22     la $a0, c_word
23
24     # So ki tu cua tu
25     addi $a1, $zero, 9
26
27     jal cyclone_word
28     nop
29
30     beq $v0, $zero, false_answer
31
32 true_answer:
33     li $v0, 55
34     la $a0, true
35     syscall
36     j end_main
37
38 false_answer:
39     li $v0, 55
40     la $a0, false
41     syscall
42
43 end_main:
44     li $v0, 10
45     syscall
46
47 #-----
48 # @cyclone_word: Kiem tra xem co phai la cyclone word hay khong
```

```

49 #-----
50
51 cyclone_word:
52 # If length of word = 0 or = 1 => TRUE
53 beq $a1, $zero, return_true
54 addi $t0, $zero, 1
55 beq $a1, $t0, return_true
56
57 addi $t0, $zero, 0      # i: 0
58 sub $t1, $a1, 1        # j: n - 1
59 loop:
60 slt $t2, $t0, $t1      # if i >= j => return 0
61 beq $t2, $zero, return_true
62
63 add $s1, $a0, $t0
64 lb $s1, 0($s1)         # c_word[i]
65 add $s2, $a0, $t1
66 lb $s2, 0($s2)         # c_word[j]
67 sle $t3, $s1, $s2      # if c_wrod[i] > c_word[j] false => 0
68 beq $t3, $zero, return_false # if return 0
69
70 addi $t4, $t0, 1
71 add $t4, $a0, $t4
72 lb $s3, 0($t4)         # c_word[i+1]
73 sle $t5, $s2, $s3      # if c_word[j] > c_word[i+1] false => 0
74 beq $t5, $zero, return_false
75
76 addi $t0, $t0, 1      # i = i + 1
77 addi $t1, $t1, -1     # j = j - 1
78 j loop
79 end_loop:
80
81 return_true:
82 addi $v0, $zero, 1
83 jr $ra
84 return_false:
85 addi $v0, $zero, 0
86 jr $ra
87
88 #-----
89 # END
90 #-----

```

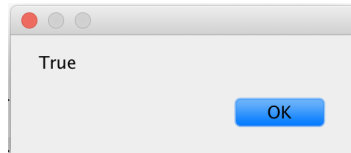
Project 24 - Cyclone Word (Challenge)

6.5 Kết quả chạy chương trình

- Đầu vào:

```
1 c_word: .asciiz "adjourned"
```

- Kết quả:



- Đầu vào:

```
1 c_word: .asciiz "settled"
```

- Kết quả:

