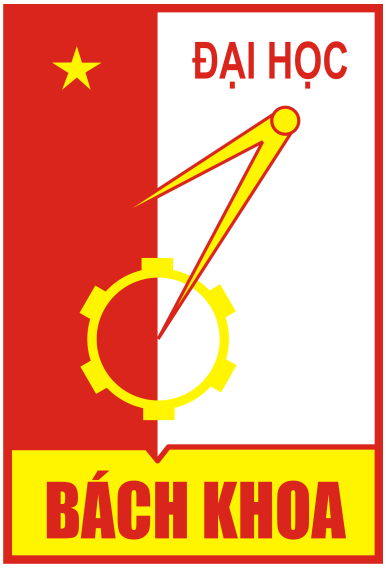
ĐẠI HỌC BÁCH KHOA HÀ NỘI

TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG



BÁO CÁO BÀI TẬP LỚN

THỰC HÀNH KIẾN TRÚC MÁY TÍNH IT3280

Học kì 20232 - Năm học: 2023 - 2024

Giảng viên hướng dẫn: ThS. Lê Bá Vui

Danh sách thành viên: Nguyễn Lương Hoàng Tùng - 20226129
Ôn Quang Tùng - 20226096

Mã lớp: 147789

Hà Nội, 2024

A. Phân công công việc

1. Nguyễn Lương Hoàng Tùng
   - MSSV: 20226129
   - Email: tung.nlh226129@sis.hust.edu.vn
   - Bài 9: Vẽ hình bằng ký tự ASCII
2. Ôn Quang Tùng
   - MSSV: 20226096
   - Email: tung.oq226096@sis.hust.edu.vn
   - Bài 5: Biểu thức trung tố hậu tố

B. Thực hiện công việc

I. Bài 9: Vẽ hình bằng ký tự ASCII

Đề bài:

Cho hình ảnh đã được chuyển thành các kí tự ASCII như hình vẽ. Đây là hình của chữ DCE có viền * và màu là các con số

```
                                          *************
**************                           *3333333333333*
*222222222222*                           *33333********
*22222******222222*                      *33333*
*22222*      *22222*                     *33333********
*22222*       *22222*      *************  *3333333333333*
*22222*       *22222*    **11111*****111* *33333********
*22222*       *22222*  **1111**       **  *33333*
*22222*      *222222*  *1111*             *33333********
*22222*******222222*  *11111*             *3333333333333*
*2222222222222222*    *11111*              *************
***************       *11111*
      ---              *1111**
    / o o \             *1111****   *****
    \   > /             **111111***111*
     -----                ***********    dce.hust.edu.vn
```

- Hãy hiển thị hình ảnh trên lên giao diện console (hoặc giao diện Display trong công cụ giả lập Keyboard and Display MMIO Simulator).
- Hãy sửa ảnh để các chữ cái DCE chỉ còn lại viền, không còn màu số ở giữa, và hiển thị.
- Hãy sửa ảnh để hoán đổi vị trí của các chữ, thành ECD, và hiển thị. Để đơn giản, các hoạ tiết đính kèm cũng được phép di chuyển theo.

- Hãy nhập từ bàn phím kí tự màu cho chữ D, C, E, rồi hiển thị ảnh trên với màu mới.

Thực hiện yêu cầu:

1. Khởi tạo biến lưu giá trị hình ảnh

Ta coi hình vẽ là một mảng hai chiều kích thước 64*16 được trình bày dưới dạng mảng một chiều kích thước 64*16=1024 kí tự. Trong đó ở mỗi dòng có: 21 ký tự cho chữ D, 21 ký tự cho chữ C, 21 ký tự cho chữ E, 1 ký tự cuối cho "\n".

Giá trị của biến String là:

```
"                          *************     \n**************
*3333333333333*    \n*222222222222222*             *33333********
\n*22222******222222*             *33333*        \n*22222*    *22222*
*33333********    \n*22222*    *22222*    ************ *3333333333333*
\n*22222*    *22222*   **11111*****111* *33333********    \n*22222*
*22222* **1111**    ** *33333*        \n*22222*    *222222* *1111*
*33333********    \n*22222*******222222* *11111*       *3333333333333*
\n*2222222222222222*  *11111*        *************    \n***************
*11111*                   \n   ---         *1111**                  \n  / o o \\
*1111****  *****             \n  \\  > /         **111111***111*            \n
-----           ***********  dce.hust.edu.vn    \n"
```

```
.data
        String: .asciiz "                                          *************       \n**************                          *3333333333333*
         "                                  *************       \n"
         "**************                   *3333333333333*      \n"
         "*222222222222222*                *33333********       \n"
         "*22222******222222*              *33333*             \n"
         "*22222*      *22222*             *33333********      \n"
         "*22222*       *22222*      *************  *3333333333333*      \n"
         "*22222*       *22222*    **11111*****111* *33333********       \n"
         "*22222*       *22222*  **1111**       **  *33333*             \n"
         "*22222*      *222222*  *1111*            *33333********      \n"
         "*22222*******222222*  *11111*            *3333333333333*     \n"
         "*2222222222222222*    *11111*             *************      \n"
         "***************       *11111*                                \n"
         "     ---               *1111**                               \n"
         "   / o o \\              *1111****   *****                     \n"
         "   \\   > /               **111111***111*                     \n"
         "    -----                  ***********    dce.hust.edu.vn     \n"
        Menu:.asciiz "\n\n\n==========================MENU==========================\n|1. Hien thi hinh anh tren giao dien
        Message: .asciiz "\nNhap 3 ky tu tuong ung voi 3 mau moi lan luot cua D,C,E\n\n\n"
        Error: .asciiz "So khong phu hop, Xin nhap lai!\n"
```

2. Tạo biến Menu

Giá trị của biến Menu là:

"\n\n\n==========MENU==========\n

|1. Hien thi hinh anh tren giao dien |\n

|2. Hien thi hinh anh chi con lai vien, khong co mau o giua |\n

|3. Hien thi hinh anh sau khi hoan doi vi tri |\n

|4. Nhap tu ban phim ki tu mau cho chu D, C, E roi hien thi |\n

|(Nhap exit de thoat chuong trinh) |\n
==========\n\n\n"

```
25  main:
26   la $a0, Menu
27   li $v0, 4
28   syscall
29
30   li $v0, 5
31   syscall
32
33   case1:
34          addi $v1, $0, 1
35          bne $v0, $v1, case2
36          j c1
37   case2:
38          addi $v1, $0, 2
39          bne $v0, $v1, case3
40          j c2
41   case3:
42          addi $v1, $0, 3
43          bne $v0, $v1, case4
44          j c3
45   case4:
46          addi $v1, $0, 4
47          bne $v0, $v1, case5
48          j c4
49   case5:
50          addi $v1, $0, 5
51          bne $v0, $v1, default
52          j Exit
53   default:
54          j main
55
```

3. In hình vẽ

Ta thực hiện in biến String:

```
#-----------------------------------------
 c1:
        li $v0, 4
        la $a0, String
        syscall
        j main
#-----------------------------------------
```

Kết quả:

```
============================MENU============================
|1. Hien thi hinh anh tren giao dien                       |
|2. Hien thi hinh anh chi con lai vien, khong co mau o giua|
|3. Hien thi hinh anh sau khi hoan doi vi tri              |
|4. Nhap tu ban phim ki tu mau cho chu D, C, E roi hien thi|
|(Nhap exit de thoat chuong trinh)                         |
============================================================


1
                                          *************
**************                           *3333333333333*
*222222222222222*                         *33333********
*22222******222222*                       *33333*
*22222*      *22222*                      *33333********
*22222*       *22222*      *************  *3333333333333*
*22222*       *22222*    **11111*****111* *33333********
*22222*       *22222*  **1111**       **  *33333*
*22222*      *222222*  *1111*             *33333********
*22222*******222222*  *11111*             *3333333333333*
*2222222222222222*    *11111*              *************
***************       *11111*
     ---               *1111**
   / o o \              *1111****   *****
   \   > /              **111111***111*
    -----                 ***********    dce.hust.edu.vn
```

4. In hình vẽ chỉ có viền

Duyệt từng kí tự trong String và in ra. Nếu kí tự được duyệt là chữ số (Có giá trị ASCII là 48 – 57) thì thay thế nó thành kí tự Space (Có giá trị ASCII là 32) rồi in ra.

```
#----------------------------------------
 c2:
        la $s0, String # luu dia chi co so cua String
        li $s1, 0       # con tro cua String
 loop2:
        beq $s1, 1024, main # Ket thuc khi in du ky tu
        lb $t0, 0($s0)       # $t0 luu gia tri phan tu trong String
        bgt $t0, 57, print2
        bgt $t0, 47, digit2  # Neu la 0->9 (ma ascii 48->57) thi thay so bang ky tu space (32)
        j print2
 digit2:
        li $t0, 32
 print2:
        li $v0, 11          # In tung ky tu
        move $a0, $t0
        syscall
        addi $s0, $s0, 1    # Tang dia chi String
        addi $s1, $s1, 1    #Tang gia tri con tro
        j loop2
#----------------------------------------
```

Kết quả:

```
==============================MENU==============================
|1. Hien thi hinh anh tren giao dien                           |
|2. Hien thi hinh anh chi con lai vien, khong co mau o giua|
|3. Hien thi hinh anh sau khi hoan doi vi tri                  |
|4. Nhap tu ban phim ki tu mau cho chu D, C, E roi hien thi|
|(Nhap exit de thoat chuong trinh)                             |
================================================================



2
                                                 *************
***************                                 *               *
*                 *                             *      ********
*      ******       *                           *      *
*      *       *      *                         *      ********
*      *         *      *       **************  *               *
*      *         *      *     **      ******    * *      ********
*      *         *      *   **     **         **  *      *
*      *       *        *   *     *              *      ********
*      *******       *   *      *               *               *
*                  *     *      *                *************
****************         *      *
     ---                  *     **
   / o o \                  *     ****    ******
   \   > /                   **       ***    *
    -----                     ************    dce.hust.edu.vn
```

5. Hoán đổi vị trí các chữ

Ở mỗi dòng ta sẽ in ra lần lượt:

21 ký tự chữ E -> 21 ký tự chữ C -> 21 ký tự chữ D -> ký tự "\n"

```
#----------------------------------------
c3:
        la $s0, String
        li $s2, -1        # Luu tung do cua con tro vao $s2
loop3j:
        addi $s2, $s2, 1
        beq $s2, 16, main    # Ket thuc khi duyet du 16 dong
        sll $s1, $s2, 6      # $s1 = $s2*64
        add $s1, $s1, $s0    # $s1 la dia chi dau moi dong cua hinh ve

        # In E
        addi $s3, $s1, 42    # Chu E bat dau tu ky tu 42 tren dong
        jal print21char

        # In C
        addi $s3, $s1, 21    # Chu C bat dau tu ky tu 21 tren dong
        jal print21char

        # In D
        addi $s3, $s1, 0     # Chu D bat dau tu dau dong
        jal print21char

        # In \n
        lb $t0, 63($s1)
        li $v0, 11          # In ky tu
        move $a0, $t0
        syscall

        j loop3j
```

```
# Ham in 21 ky tu
print21char:
        li $t4, 0
        loop3i:
                lb $t0, 0($s3)    # $t0 luu gia tri phan tu trong String
                li $v0, 11        # In ky tu
                move $a0, $t0
                syscall

                addi $s3, $s3, 1 # $s3 ++
                addi $t4, $t4, 1 # $t4 ++
                bne $t4, 21, loop3i
                jr $ra
#----------------------------------------
```

Kết quả:

```
==============================MENU============================
|1. Hien thi hinh anh tren giao dien                         |
|2. Hien thi hinh anh chi con lai vien, khong co mau o giua|
|3. Hien thi hinh anh sau khi hoan doi vi tri                |
|4. Nhap tu ban phim ki tu mau cho chu D, C, E roi hien thi|
|(Nhap exit de thoat chuong trinh)                           |
==============================================================


3
 *************
*3333333333333*                          **************
*33333********                           *222222222222222*
*33333*                                  *22222******222222*
*33333********                           *22222*       *22222*
*3333333333333*             *************  *22222*       *22222*
*33333********            **11111*****111* *22222*       *22222*
*33333*                 **1111**        **  *22222*       *22222*
*33333********          *1111*              *22222*      *222222*
*3333333333333*        *11111*              *22222*******222222*
 *************         *11111*              *2222222222222222*
                       *11111*              ***************
                        *1111**                  ---
                         *1111****   *****    / o o \
                           **111111***111*     \   > /
dce.hust.edu.vn              **********         -----
```

6. Đổi màu chữ

- Lần lượt nhập các chữ số vào để thu được màu cho 3 chữ D,C,E.
- Nếu phát hiện không phải chữ số thì yêu cầu nhập lại.
- Lưu giá trị ASCII của số tương ứng vào các thanh ghi $t7, $t8, $t9.

```
125 #---------------------------------------
126  c4:
127   Input:
128         li $v0, 4
129         la $a0, Message
130         syscall
131
132         li $v0, 12
133         syscall
134
135         bgt $v0, 57, error
136         blt $v0, 48, error
137         addi $t7, $v0, 0  # $t7 chua mau chu D
138
139         li $v0, 12
140         syscall
141
142         bgt $v0, 57, error
143         blt $v0, 48, error
144         addi $t8, $v0, 0  # $t8 chua mau chu C
145
146         li $v0, 12
147         syscall
148
149         bgt $v0, 57, error
150         blt $v0, 48, error
151         addi $t9, $v0, 0  # $t9 chua mau chu E
152         j Inputend
153
154   error:
155         la $a0, Error
156         li $v0, 4
157         syscall
158         j Input
```

Xử lý dữ liệu:

- Duyệt trên từng dòng một và in lần lượt: 21 ký tự D -> 21 ký tự C -> 21 ký tự E -> ” \n”

- Trong quá trình in các ký tự, nếu phát hiện chữ số (mã ASCII 48-57) thì ta thay thế chúng bằng giá trị mới được lưu trong $t7 với chữ D, $t8 với chữ C, $t9 với chữ E

```
Inputend:
      la $a0, newline
      li $v0, 4
      syscall
      la $s0, String
      li $s2, -1  # $s2 la tung do con tro
loop4j:
      addi $s2, $s2, 1
      beq $s2, 16, main # Ket thuc khi duyet du 16 dong
      sll $s1, $s2, 6   # $s1 = $s2 * 64
      add $s1, $s1, $s0 # $s1 la dia chi dau moi dong cua hinh ve

      addi $s3, $s1, 0  # Chu D bat dau tu dau dong
      move $t6, $t7
      jal print21charc4

      addi $s3, $s1, 21 # Chu C bat dau tu ky tu 21 tren dong
      move $t6, $t8
      jal print21charc4

      addi $s3, $s1, 42 # Chu E bat dau tu ky tu 42 tren dong
      move $t6, $t9
      jal print21charc4

      lb $t0, 63($s1)
      li $v0, 11        # In ky tu
      move $a0, $t0
      syscall
      j loop4j
print21charc4:
      li $t4, 0
```

```
 loop4i:
      lb $t0, 0($s3)          # $t0 luu gia tri phan tu trong String
      bgt $t0, 57, print4
      bgt $t0, 47, digit4     # Cac chu so 0 -> 9 co ma ascii 48 -> 57
      j print4
 digit4:
      move $t0, $t6           # Doi ky tu ban dau thanh ky tu moi nhap
 print4:
      li $v0, 11              # In ky tu
      move $a0, $t0
      syscall

      addi $s3, $s3, 1        # $s3 ++
      addi $t4, $t4, 1        # $t4 ++
      bne $t4, 21, loop4i
      jr $ra
Exit:
```

Kết quả:

```
============================MENU============================
|1. Hien thi hinh anh tren giao dien                       |
|2. Hien thi hinh anh chi con lai vien, khong co mau o giua|
|3. Hien thi hinh anh sau khi hoan doi vi tri              |
|4. Nhap tu ban phim ki tu mau cho chu D, C, E roi hien thi|
|(Nhap exit de thoat chuong trinh)                         |
============================================================


4

Nhap 3 ky tu tuong ung voi 3 mau moi lan luot cua D,C,E


689
                                            *************
**************                             *9999999999999*
*666666666666666*                           *99999********
*66666******666666*                         *99999*
*66666*      *66666*                        *99999********
*66666*       *66666*      *************   *9999999999999*
*66666*       *66666*    **88888*****888*  *99999********
*66666*       *66666*  **8888**       **   *99999*
*66666*      *666666*  *8888*              *99999********
*66666*******666666*  *88888*              *9999999999999*
*6666666666666666*    *88888*               *************
***************       *88888*
     ---               *8888**
   / o o \              *8888****   *****
   \   > /               **888888***888*
    -----                  **********    dce.hust.edu.vn
```

```
============================MENU============================
|1. Hien thi hinh anh tren giao dien                       |
|2. Hien thi hinh anh chi con lai vien, khong co mau o giua|
|3. Hien thi hinh anh sau khi hoan doi vi tri              |
|4. Nhap tu ban phim ki tu mau cho chu D, C, E roi hien thi|
|(Nhap exit de thoat chuong trinh)                         |
============================================================


4

Nhap 3 ky tu tuong ung voi 3 mau moi lan luot cua D,C,E


12a
So khong phu hop, Xin nhap lai!
```

II. Bài 5: Biểu thức trung tố hậu tố

1. Đề bài:

Viết chương trình tính giá trị biểu thức bất kỳ bằng phương pháp duyệt biểu thức hậu tố.

Các yêu cầu cụ thể:

1. Nhập vào biểu thức trung tố, ví dụ: 9 + 2 + 8 * 6
2. In ra biểu thức ở dạng hậu tố, ví dụ: 9 2 + 8 6 * +
3. Tính ra giá trị của biểu thức vừa nhập

Các hằng số là số nguyên, trong phạm vi từ 0 → 99.

Toán tử bao gồm các phép toán cộng, trừ, nhân, chia lấy thương (/), chia lấy dư (%), đóng mở ngoặc.

2. Thực hiện yêu cầu:

- *Phân tích thuật toán:*

Để chuyển từ biểu thức trung tố sang biểu thức hậu tố, ta đọc lần lượt từng ký tự trong biểu thức trung tố, giả sử ký tự đọc được là c:

- Nếu c là số -> thêm vào biểu thức hậu tố +
- Nếu c là toán tử -> thực hiện thêm toán tử vào stack. Nếu stack rỗng, thêm c vào stack, nếu stack không rỗng, so sánh thứ tự ưu tiên của c với toán tử ở đỉnh stack, nếu độ ưu tiên của c <= độ ưu tiên của toán tử ở đỉnh stack, lấy toán tử ở đỉnh ra và thêm vào biểu thức hậu tố, lặp lại việc so sánh trên cho đến khi độ ưu tiên của c >= độ ưu tiên của toán tử ở đỉnh stack, sau đó thêm c vào stack.
- + Nếu c là toán tử (-> thêm luôn c vào stack. +
- Nếu c là toán tử) -> lấy lần lượt các toán tử trong stack ra và thêm vào biểu thức hậu tố cho đến khi gặp toán tử) . Lưu ý, không thêm toán tử) vào trong biểu thức hậu tố. +
- Khi duyệt hết biểu thức trung tố, nếu còn toán tự trong stack, thực hiện lấy lần lượt các toán tử ra và thêm vào biểu thức trung tố cho đến khi stack rỗng.

Vì các toán hạng có giá trị từ 0 -> 99 nên khi thêm vào biểu thức hậu tố, cần thêm các dấu cách để ngăn cách giữa các phần tử.

Để tính giá trị biểu thức hậu tố:

- Thực hiện duyệt biểu thức hậu tố, nếu gặp toán hạng thì thêm toán hạng vào stack, nếu gặp toán tử, lấy hai toán hạng trong stack ra và tính toán phép tính theo toán tử tương ứng, kết quả của phép tính được lưu trở lại vào trong stack.
- Lặp lại cho đến khi duyệt hết biểu thức hậu tố, kết quả cuối cùng được lưu trong stack. 10

Thứ tự ưu tiên của các toán tử:

Toán tử	Độ ưu tiên
(	0
+	1
-	1
*	2
/	2
%	2
)	3

- *Menu cho người dùng:*

```
.data
        askMessage: .asciiz "Continue?\n1. Press 1 to continue\n2. Press 2 to end programme\n"
        endMessage: .asciiz "\nGoodnight! Sweet dream!"
        errorMessage: .asciiz "\nInvalid input!"
        infixMessage: .asciiz "Infix expression: "
        postfixMessage: .asciiz "Postfix expression: "
        inputMessage: .asciiz "Enter infix: "
        resultMessage: .asciiz "Calculated result: "
        startMessage: .asciiz "\nPlease enter infix expression\nNote: only allowed to use + - * / % ()\nNumber from 00-99\n"

        infix: .space 256
        postfix: .space 256
        operator: .space 256
        converter: .word 1
        covertToFloat: .word 1
        stack: .float
```

```
.text
start:
# Get infix expression
        li $v0, 4
        la $a0, startMessage
        syscall
        li $v0, 4
        la $a0, inputMessage
        syscall
        li $v0, 8
        la $a0, infix
        la $a1, 256
        syscall
```

```
ask:                    # Ask user to continue or not
        li $v0, 4
        la $a0, askMessage
        syscall
        li $v0, 12
        syscall
        beq $v0,'1', start
        beq $v0,'2', end
```

```
Please enter infix expression
Note: only allowed to use + - * / % ()
Number from 00-99
Enter infix:
```

```
Infix expression: 12%(34+56)-78

Invalid input!
Continue?
1. Press 1 to continue
2. Press 2 to end programme
```

Clear

- *Lấy input và in ra infix:*

```
.text
start:
# Get infix expression
        li $v0, 4
        la $a0, startMessage
        syscall
        li $v0, 4
        la $a0, inputMessage
        syscall
        li $v0, 8
        la $a0, infix
        la $a1, 256
        syscall

# Print infix
        li $v0, 4
        la $a0, infixMessage
        syscall
        li $v0, 4
        la $a0, infix
        syscall
```

```
Please enter infix expression
Note: only allowed to use + - * / % ()
Number from 00-99
Enter infix: 12+34*56
Infix expression: 12+34*56
```

- *Khởi tạo các thanh ghi trước khi duyệt qua infix:*

```
# Status
        li $s7,0                # 0 = initially receive nothing
                                # 1 = receive number
                                # 2 = receive operator
                                # 3 = receive (
                                # 4 = receive )

        li $t9,0                # Count digit
        li $t5,-1               # Postfix top offset
        li $t6,-1               # Operator top offset
        la $t1, infix           # Infix current byte address +1 each loop
        la $t2, postfix
        la $t3, operator
        addi $t1,$t1,-1         # Set initial address of infix to -1

```

- *Bắt đầu duyệt infix, xét các trường hợp token:*

```
scanInfix:                  # Loop for each character in postfix
# Check all valid input option
        addi $t1,$t1,1              # Increase infix position
        lb $t4, ($t1)               # Load current infix input
        beq $t4, ' ', scanInfix     # If scan spacebar ignore and scan again
        beq $t4, '\n', EOF          # Scan end of input --> pop all operator to postfix
        beq $t9,0,digit1            # If state is 0 digit
        beq $t9,1,digit2            # If state is 1 digit
        beq $t9,2,digit3            # If state is 2 digit
        continueScan:
        beq $t4, '+', plusMinusOperator
        beq $t4, '-', plusMinusOperator
        beq $t4, '*', multiplyDivideOperator
        beq $t4, '/', multiplyDivideOperator
        beq $t4, '%', modulusOperator
        beq $t4, '(', openBracket
        beq $t4, ')', closeBracket
```

```
finishPrint:
        li $v0, 11
        li $a0, '\n'
        syscall
```

- *Kết thúc duyệt infix, đồng thời thực hiện in ra postfix:*

```
finishScan:
# Print postfix expression
        # Print prompt:
        li $v0, 4
        la $a0, postfixMessage
        syscall
        li $t6,-1               # Load current of Postfix offset to -1

printPost:
        addi $t6,$t6,1          # Increment current of Postfix offset
        add $t8,$t2,$t6         # Load address of current Postfix
        lbu $t7,($t8)           # Load value of current Postfix
        bgt $t6,$t5,finishPrint# Print all postfix --> calculate
        bgt $t7,99,printOp      # If current Postfix > 99 --> an operator
        # If not then current Postfix is a number
        li $v0, 1
        add $a0,$t7,$zero
        syscall
        li $v0, 11
        li $a0, ' '
        syscall
        j printPost             # Loop
        printOp:
        li $v0, 11
        addi $t7,$t7,-100       # Decode operator
        add $a0,$t7,$zero
        syscall
        li $v0, 11
        li $a0, ' '
        syscall
        j printPost             # Loop
```

- *Thực hiện tính phép tính sau khi có postfix. Load các biến khởi tạo, và thực hiện đẩy các toán hạng vào trong stack nếu chưa gặp toán tử nào:*

```
calPost:
        addi $t6,$t6,1          # Increment current of Postfix offset
        add $t8,$t2,$t6         # Load address of current Postfix
        lbu $t7,($t8)           # Load value of current Postfix
        bgt $t6,$t5,printResult# Calculate for all postfix --> print
        bgt $t7,99,calculate    # If current Postfix > 99 --> an operator --> popout 2 number to calculate
        # If not then current Postfix is a number
        addi $t9,$t9,4          # Current stack top offset
        add $t4,$t3,$t9         # Current stack top address
        sw $t7, 0($t4)
        j calPost               # Loop
```

- *Khi gặp toán tử, ta thực hiện lấy 2 giá trị ra khỏi stack và thực hiện phép toán:*

```
        calculate:
                # Pop 1 number
                add $t4,$t3,$t9
                lw $t0,($t4)
                # Pop next number
                addi $t9,$t9,-4
                add $t4,$t3,$t9
                lw $t1,($t4)
                # Decode operator
                beq $t7, 143, plus
                beq $t7, 145, minus
                beq $t7, 142, multiply
                beq $t7, 147, divide
                beq $t7, 137, modulus
```

```
                plus:
                        add $t0,$t0,$t1
                        sw $t0,($t4)
                        j calPost
                minus:
                        sub $t0,$t1,$t0
                        sw $t0,($t4)
                        j calPost
                multiply:
                        mul $t0,$t1,$t0
                        sw $t0,($t4)
                        j calPost
                divide:
                        div $t1, $t0
                        mflo $t0
                        sw $t0,($t4)
                        j calPost
                modulus:
                        div $t1, $t0
                        mfhi $t0
                        sw $t0,($t4)
                        j calPost               # Jump back to calPost
```

- *In ra kết quả sau khi tính:*

```
printResult:
        li $v0, 4
        la $a0, resultMessage
        syscall
        li $v0, 1
        lw $a0,($t4)
        syscall
        li $v0, 11
        li $a0, '\n'
        syscall
```

```
Calculated result: 32
```

- *Kết thúc:*

```
193 # End program
194 end:
195         li $v0, 4
196         la $a0, endMessage
197         syscall
198         li $v0, 10
199         syscall
```

- *Các chương trình con:*

```
74  wrongInput:     # When detect wrong input situation
75          li $v0, 4
76          la $a0, errorMessage
77          syscall
78          li $v0, 11
79          li $a0, '\n'
80          syscall
81          j ask
```

```
201 # Sub program
202 EOF:
203         beq $s7,2,wrongInput                # End with an operator or open bracket
204         beq $s7,3,wrongInput
205         beq $t5,-1,wrongInput               # Input nothing
206         j popAll
```

```
207 digit1:
208         beq $t4,'0',store1Digit
209         beq $t4,'1',store1Digit
210         beq $t4,'2',store1Digit
211         beq $t4,'3',store1Digit
212         beq $t4,'4',store1Digit
213         beq $t4,'5',store1Digit
214         beq $t4,'6',store1Digit
215         beq $t4,'7',store1Digit
216         beq $t4,'8',store1Digit
217         beq $t4,'9',store1Digit
218         j continueScan
```

```
220 digit2:
221         beq $t4,'0',store2Digit
222         beq $t4,'1',store2Digit
223         beq $t4,'2',store2Digit
224         beq $t4,'3',store2Digit
225         beq $t4,'4',store2Digit
226         beq $t4,'5',store2Digit
227         beq $t4,'6',store2Digit
228         beq $t4,'7',store2Digit
229         beq $t4,'8',store2Digit
230         beq $t4,'9',store2Digit
231         # If do not receive second digit
232         jal numberToPost
233         j continueScan
```

```
digit3:
        # If scan third digit --> error
        beq $t4,'0',wrongInput
        beq $t4,'1',wrongInput
        beq $t4,'2',wrongInput
        beq $t4,'3',wrongInput
        beq $t4,'4',wrongInput
        beq $t4,'5',wrongInput
        beq $t4,'6',wrongInput
        beq $t4,'7',wrongInput
        beq $t4,'8',wrongInput
        beq $t4,'9',wrongInput
        # If do not receive third digit
        jal numberToPost
        j continueScan
```

```
plusMinusOperator:                  # Input is + -
        beq $s7,2,wrongInput        # Receive operator after operator or open bracket
        beq $s7,3,wrongInput
        beq $s7,0,wrongInput        # Receive operator before any number
        li $s7,2                    # Change input status to 2
        continuePlusMinus:
        beq $t6,-1,inputToOperator          # There is nothing in Operator stack --> push into
        add $t8,$t6,$t3             # Load address of top Operator
        lb $t7,($t8)                # Load byte value of top Operator
        beq $t7,'(',inputToOperator         # If top is ( --> push into
        beq $t7,'+',equalPrecedence # If top is + -
        beq $t7,'-',equalPrecedence
        beq $t7,'*',lowerPrecedence # If top is * /
        beq $t7,'/',lowerPrecedence
        beq $t7, '%', equalPrecedence
```

```
multiplyDivideModulusOperator:              # Input is * /
        beq $s7,2,wrongInput        # Receive operator after operator or open bracket
        beq $s7,3,wrongInput
        beq $s7,0,wrongInput        # Receive operator before any number
        li $s7,2                    # Change input status to 2
        beq $t6,-1,inputToOperator          # There is nothing in Operator stack --> push into
        add $t8,$t6,$t3             # Load address of top Operator
        lb $t7,($t8)                # Load byte value of top Operator
        beq $t7,'(',inputToOperator         # If top is ( --> push into
        beq $t7,'+',inputToOperator         # If top is + - --> push into
        beq $t7,'-',inputToOperator
        beq $t7,'*',equalPrecedence # If top is * /
        beq $t7,'/',equalPrecedence
        beq $t7, '%', equalPrecedence
```

```
openBracket:                # Input is (
        beq $s7,1,wrongInput        # Receive open bracket after a number or close bracket
        beq $s7,4,wrongInput
        li $s7,3                    # Change input status to 3
        j inputToOperator
closeBracket:               # Input is )
        beq $s7,2,wrongInput        # Receive close bracket after an operator or operator
        beq $s7,3,wrongInput
        li $s7,4
        add $t8,$t6,$t3             # Load address of top Operator
        lb $t7,($t8)                # Load byte value of top Operator
        beq $t7,'(',wrongInput      # Input contain () without anything between --> error
        continueCloseBracket:
        beq $t6,-1,wrongInput       # Can't find an open bracket --> error
        add $t8,$t6,$t3             # Load address of top Operator
        lb $t7,($t8)                # Load byte value of top Operator
        beq $t7,'(',matchBracket    # Find matched bracket
        jal operatorToPostfix               # Pop the top of Operator to Postfix
        j continueCloseBracket      # Then loop again till find a matched bracket or error
```

```
equalPrecedence:    # Mean receive + - and top is + - || receive * / % and top is * / %
        jal operatorToPostfix               # Pop the top of Operator to Postfix
        j inputToOperator                   # Push the new operator in
lowerPrecedence:    # Mean receive + - and top is * / %
        jal operatorToPostfix               # Pop the top of Operator to Postfix
        j continuePlusMinus         # Loop again
inputToOperator:                    # Push input to Operator
        add $t6,$t6,1               # Increment top of Operator offset
        add $t8,$t6,$t3             # Load address of top Operator
        sb $t4,($t8)                # Store input in Operator
        j scanInfix
operatorToPostfix:                  # Pop top of Operator in push into Postfix
        addi $t5,$t5,1              # Increment top of Postfix offset
        add $t8,$t5,$t2             # Load address of top Postfix
        addi $t7,$t7,100            # Encode operator + 100
        sb $t7,($t8)                # Store operator into Postfix
        addi $t6,$t6,-1             # Decrement top of Operator offset
        jr $ra
matchBracket:               # Discard a pair of matched brackets
        addi $t6,$t6,-1             # Decrement top of Operator offset
        j scanInfix
```

```
popAll:                         # Pop all Operator to Postfix
        jal numberToPost
        beq $t6,-1,finishScan           # Operator empty --> finish
        add $t8,$t6,$t3                 # Load address of top Operator
        lb $t7,($t8)                    # Load byte value of top Operator
        beq $t7,'(',wrongInput          # Unmatched bracket --> error
        beq $t7,')',wrongInput
        jal operatorToPostfix
        j popAll                        # Loop till Operator empty
store1Digit:
        beq $s7,4,wrongInput            # Receive number after )
        addi $s4,$t4,-48                # Store first digit as number
        add $t9,$zero,1                 # Change status to 1 digit
        li $s7,1
        j scanInfix
store2Digit:
        beq $s7,4,wrongInput            # Receive number after )
        addi $s5,$t4,-48                # Store second digit as number
        mul $s4,$s4,10
        add $s4,$s4,$s5                 # Stored number = first digit * 10 + second digit
        add $t9,$zero,2                 # Change status to 2 digit
        li $s7,1
        j scanInfix
```

```
numberToPost:
        beq $t9,0,endnumberToPost
        addi $t5,$t5,1
        add $t8,$t5,$t2
        sb $s4,($t8)                    # Store number in Postfix
        add $t9,$zero,$zero             # Change status to 0 digit
        endnumberToPost:
        jr $ra
```