Phòng thí nghiệmBài tập5

CharacterstringwithSYSCALL chức năng và sắp xếp

Bàn thắng

Sau bài thực hành trong phòng thí nghiệm này, bạn sẽ hiểu cơ chế lưu trữ

Chuỗi ASCII và Unicode. Bạn sẽ có thể lập trình để xử lý chuỗi và đặt

chuỗi vào bảng điều khiển. Ngoài ra, bạn nên biết cách sắp xếp danh sách các phần tử.

Văn chương

Patterson, Henessy (COD): phần 2.8, 2.13

Sự chuẩn bị

Trước khi bắt đầu bài thực hành, bạn nên xem lại sách giáo khoa, phần 6.1 và đọc kỹ phòng thí nghiệm này. Bạn cũng nên đọc Tài liệu tham khảo về môi trường phòng thí nghiệm Mips để tìm cách sử dụng các thủ tục printf, putchar. , v.v.

Về SYSCALL

Một số dịch vụ hệ thống, chủ yếu cho đầu vào và đầu ra, có sẵn để sử dụng bởi chương trình MIPS của bạn. Chúng được mô tả trong bảng dưới đây.

Nội dung thanh ghi MIPS không bị ảnh hưởng bởi lệnh gọi hệ thống, ngoại trừ các thanh ghi kết quả như được chỉ định trong bảng dưới đây.

HowtouseSYSCALLsystemservices 1. Nạp số dịch vụ vào thanh ghi $ v0.

2. Tải các giá trị đối số, nếu có, trong $ a0, $ a1, $ a2 hoặc $ f12 như đã chỉ định.
3. Đưa ra lệnh SYSCALL.
4. Lấy các giá trị trả về, nếu có, từ các thanh ghi kết quả như được chỉ định. 5.

Ví dụ: hiển thị một giá trị số nguyên trong bảng điều khiển

```
    li $ v0, 1 # dịch vụ 1 là số nguyên in
    li $ a0, 0x307 # interger sẽ được in là 0x307
    syscall                         # hành hình
```

TableofFrequentlyAvailableServices

Dịch vụ	Mã trong $ v0 1	Tranh luận	Kết quả
in chuỗi số		$ a0 = số nguyên để	
nguyên	4	in $ a0 = địa chỉ của chuỗi kết thúc rỗng để in	
đọc số nguyên	5		$ v0 chứa số nguyên được đọc

đọc chuỗi	ảnh	$ a0 = địa chỉ của bộ đệm đầu vào $ a1 = số ký tự tối đa để đọc	Xem ghi chú bên dưới bảng
lối ra	10	(chấm dứt thực hiện)	
ký tự in	11	$ a0 = ký tự để in	Xem ghi chú bên dưới bảng
đọc ký tự mở tệp	12		$ v0 chứa ký tự đọc $ v0 chứa
	13	$ a0 = địa chỉ của chuỗi kết thúc rỗng chứa tên tệp $ a1 = cờ $ a2 = mode	bộ mô tả tệp (âm nếu lỗi). Xem ghi chú bên dưới bảng
đọc từ tệp	14	$ a0 = trình mô tả tệp $ a1 = địa chỉ của bộ đệm đầu vào $ a2 = số ký tự tối đa để đọc $ a0 = trình mô tả tệp	$ v0 chứa số ký tự được đọc (0 nếu cuối tệp, âm nếu lỗi). Xem ghi chú bên dưới bảng
ghi vào tập tin	15	$ a1 = địa chỉ của bộ đệm đầu ra $ a2 = số ký tự cần viết $ a0 = bộ	$ v0 chứa số ký tự được viết (âm nếu lỗi). Xem ghi chú bên dưới bảng
Đóng tập tin	16	mô tả tệp $ a0 = kết	
exit2 (kết thúc với giá trị) thời gian (thời gian	17	quả kết thúc	Xem ghi chú bên dưới bảng
hệ thống)	30		$ a0 = bậc thấp 32 bit thời gian hệ thống $ a1 = bậc cao 32 bit của thời gian hệ thống. Xem ghi chú bên dưới bảng
MIDI ra	31	$ a0 = sân (0-127) $ a1 = thời lượng tính bằng mili giây $ a2 = công cụ (0-127) $ a3 = volume (0-127) $	Tạo ra âm báo và trở lại ngay lập tức. Xem ghi chú bên dưới bảng
ngủ	32	a0 = thời gian ngủ tính bằng mili giây.	Làm cho chuỗi MARS Java ở chế độ ngủ trong (ít nhất) số mili giây được chỉ định. Thời gian này sẽ không chính xác, vì việc triển khai Java sẽ thêm một số chi phí.
MIDI ra đồng bộ	33	$ a0 = sân (0-127) $ a1 = thời lượng tính bằng mili giây $ a2 = công cụ (0-127) $ a3 = volume (0-127) $	Tạo âm sắc và trở lại sau khi hoàn thành giai điệu. Xem ghi chú bên dưới bảng
in số nguyên trong hệ thập lục phân	34	a0 = số nguyên để in	Giá trị được hiển thị là 8 chữ số thập lục phân, phần đệm bên trái với các số 0 nếu cần thiết.
in số nguyên trong hệ nhị phân	35	$ a0 = số nguyên để in	Giá trị được hiển thị là 32 bit, đệm bên trái bằng số 0 nếu cần. $ a0 = số
in số nguyên dưới dạng không dấu	36	nguyên để in	Được hiển thị dưới dạng giá trị thập phân không dấu.
đặt hạt giống	40	$ a0 = id của trình tạo số giả ngẫu nhiên (int bất kỳ).	Không có giá trị nào được trả lại. Đặt hạt giống của trình tạo số giả ngẫu nhiên Java cơ bản tương ứng

		$ a1 = hạt giống cho trình tạo số giả ngẫu nhiên tương ứng. $ a0 = id	(java.util.Random). Xem ghi chú bên dưới bảng
int ngẫu nhiên	41	của trình tạo số giả ngẫu nhiên (int bất kỳ).	$ a0 chứa giá trị int giả tiếp theo, được phân phối đồng nhất từ giá trị này trình tự của trình tạo số ngẫu nhiên. Xem ghi chú bên dưới bảng
phạm vi int ngẫu nhiên	42	$ a0 = id của trình tạo số giả ngẫu nhiên (int bất kỳ). $ a1 = giới hạn trên của dải giá trị được trả về. $ a0 = địa chỉ của chuỗi kết	$ a0 chứa giá trị int ngẫu nhiên, được phân phối đồng đều trong phạm vi 0 = [int] [giới hạn trên], được rút ra từ trình tự của trình tạo số ngẫu nhiên này. Xem ghi chú bên dưới bảng
ConfirmDialog	50	thúc null là thông báo cho người dùng	$ a0 chứa giá trị của tùy chọn do người dùng chọn 0: Có 1: Không 2: Hủy $ a0
InputDialogInt	51	$ a0 = địa chỉ của chuỗi kết thúc null là thông báo cho người dùng	chứa int read $ a1 chứa giá trị trạng thái 0: Trạng thái OK -1: dữ liệu đầu vào không thể được phân tích cú pháp chính xác -2: Hủy đã được chọn -3: OK đã được chọn nhưng không có dữ liệu nào được nhập vào trường Xem ghi chú Dịch
InputDialogString	54	$ a0 = địa chỉ của chuỗi kết thúc null là thông báo cho người dùng $ a1 = địa chỉ của bộ đệm đầu vào $ a2 = số ký tự tối đa để đọc	vụ 8 bên dưới bảng $ a1 chứa giá trị trạng thái 0: Trạng thái OK. Bộ đệm chứa chuỗi đầu vào. -2: Hủy đã được chọn. Không thay đổi bộ đệm. -3: OK đã được chọn nhưng không có dữ liệu nào được nhập vào trường. Không thay đổi bộ đệm. -4: độ dài của chuỗi đầu vào vượt quá mức tối đa được chỉ định. Bộ đệm chứa chuỗi đầu vào tối đa cho phép cộng với giá trị rỗng kết thúc.
MessageDialog	55	$ a0 = địa chỉ của chuỗi kết thúc null là thông báo cho người dùng $ a1 = loại thông báo sẽ được hiển thị: 0: thông báo lỗi, được biểu thị bằng biểu tượng Lỗi 1: thông báo thông tin, được biểu thị bằng Biểu tượng thông tin 2: thông báo cảnh báo, được biểu thị bằng Biểu tượng cảnh báo 3: thông báo câu hỏi, được biểu thị bằng biểu tượng Câu hỏi	N / A

		khác: thông báo thuần túy (không có biểu	
MessageDialogInt	56	tượng hiển thị) $ a0 = địa chỉ của chuỗi kết thúc rỗng là thông báo loại thông tin cho người dùng $ a1 = giá trị int để hiển thị ở dạng chuỗi sau chuỗi đầu	N / A
MessageDialogString	59	tiên $ a0 = địa chỉ của chuỗi kết thúc null là một thông báo kiểu thông tin cho người dùng $ a1 = địa chỉ của chuỗi kết thúc null để hiển thị sau chuỗi đầu tiên	N / A

1. printinteger

in một số nguyên ra đầu ra tiêu chuẩn (bảng điều khiển).

Tranh luận):

$ v0 = 1

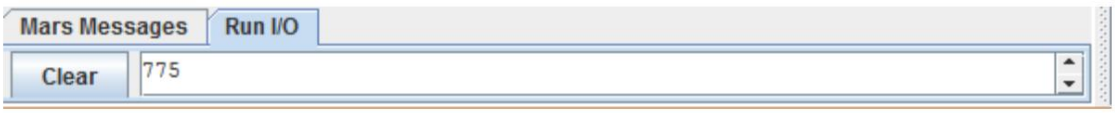
$ a0 = số sẽ được in

Giá trị trả lại:

không ai

Thí dụ:

```
    li $ v0, 1 li                # dịch vụ 1 là số nguyên in
    $ a0, 0x307 syscall          # interger sẽ được in là 0x307
    và kết quả là                # hành hình
```



2. MessageDialogInt

hiển thị một số nguyên cho hộp thoại thông báo kiểu thông tin.

Tranh luận):

$ v0 = 56

$ a0 = địa chỉ của chuỗi thông báo kết thúc bằng null

$ a1 = giá trị int

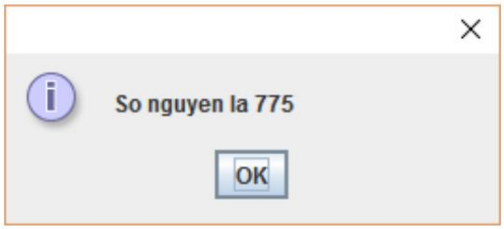
Giá trị trả lại:

không ai

Thí dụ:

```
 .dữ liệu
Tin nhắn: .asciiz "So nguyen la"
 .chữ
    li $ v0, 56 la
    $ a0, Tin nhắn
    li $ a1, cuộc gọi            # interger sẽ được in là 0x307
    tổng hợp 0x307               # hành hình
```

và kết quả là



3. chuỗi in

Bản in được định dạng thành đầu ra tiêu chuẩn (bảng điều khiển).

Tranh luận):

$ v0 = 4

$ a0 = giá trị được in

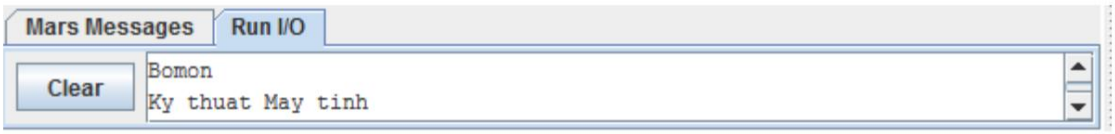
Giá trị trả lại:

không ai

Thí dụ:

```
.dữ liệu
Tin nhắn: .asciiz "Bomon \ nKy thuat May tinh"
.chữ
    li $ v0, 4
    la $ a0, Tin nhắn
    syscall
```

và kết quả là



4. MessageDialogString

Hiển thị một chuỗi cho hộp thoại thông báo kiểu thông tin

Tranh luận):

$ v0 = 59

$ a0 = địa chỉ của chuỗi thông báo kết thúc bằng null

$ a1 = địa chỉ của giá trị chuỗi bị kết thúc bằng null

Giá trị trả lại:

không ai

Thí dụ:

```
.dữ liệu
Tin nhắn: .asciiz "Bomon \ nKy thuat May tinh:"
Địa chỉ: .asciiz "phong 502, B1"
.chữ
    li $ v0, 59
    la $ a0, Tin nhắn
    la $ a1, Cuộc gọi
    syscall địa chỉ và kết
```

quả là

5. readinteger

Nhận một số nguyên từ đầu vào chuẩn (bàn phím).

Tranh luận):

$ v0 = 5

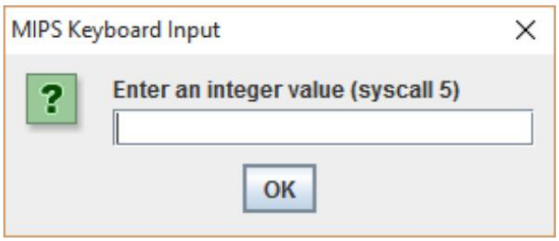
Giá trị trả lại:

$ v0 = chứa số nguyên được đọc

Thí dụ:

```
li $ v0, 5
syscall
```

và kết quả là



6. InputDialogInt

Hiển thị hộp thoại thông báo để đọc một số nguyên với trình phân tích cú pháp nội dung

Tranh luận):

$ v0 = 51

$ a0 = địa chỉ của chuỗi thông báo kết thúc bằng rỗng

Giá trị trả lại:

$ a0 = chứa int read

$ a1 = chứa giá trị trạng thái

0: Trạng thái OK

-1: dữ liệu đầu vào không thể được phân tích cú pháp chính xác

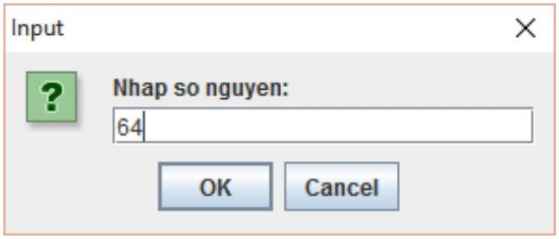
-2: Hủy đã được chọn

-3: OK đã được chọn nhưng không có dữ liệu nào được nhập vào trường

Thí dụ:

```
.dữ liệu
Tin nhắn: .asciiz "Nhap so nguyen:"
.chữ
    li $ v0, 51
    la $ a0, Tin nhắn
    syscall
```

và kết quả là



7. đọc chuỗi

Lấy một chuỗi từ đầu vào chuẩn (bàn phím).

Tranh luận):

$ v0 = 8

$ a0 = địa chỉ của bộ đệm đầu vào
$ a1 = số ký tự tối đa để đọc

Giá trị trả lại:

không ai

Nhận xét:

Đối với độ dài được chỉ định n ($ a1), chuỗi không được dài hơn n-1.

- Nếu ít hơn, hãy thêm dòng mới để kết thúc.
- Trong cả hai trường hợp, sau đó đệm bằng byte rỗng

Chỉ trong những trường hợp đặc biệt:

Nếu n = 1, đầu vào bị bỏ qua và byte trống được đặt tại địa chỉ bộ đệm.

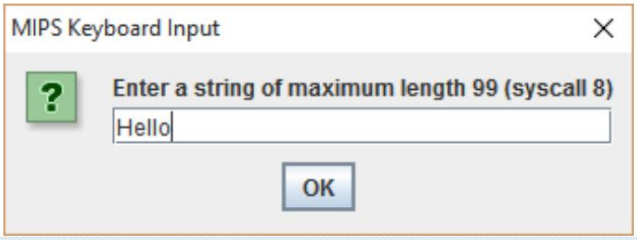
Nếu n <1, đầu vào bị bỏ qua và không có gì được ghi vào bộ đệm.

Thí dụ:

```
.dữ liệu
Tin nhắn: .space 100 .text        # Buffer 100 byte chua chuoi ki tu can

    li $ v0, 8
    la $ a0, Tin nhắn
    li $ a1, 100
    syscall
```

và kết quả là



Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)
0x10010000	l l e H	\0 \0 \n o	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0
0x10010020	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0
0x10010040	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0
0x10010060	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0

8. InputDialogString

Hiển thị hộp thoại thông báo để đọc một chuỗi với trình phân tích cú pháp nội dung

Tranh luận):

$ v0 = 54

$ a0 = địa chỉ của chuỗi thông báo kết thúc bằng rỗng

$ a1 = địa chỉ của bộ đệm đầu vào

$ a2 = số ký tự tối đa để đọc

Giá trị trả lại:

$ a1 chứa giá trị trạng thái

0: Trạng thái OK

-2: Hủy đã được chọn. Không thay đổi bộ đệm.

-3: OK đã được chọn nhưng không có dữ liệu nào được nhập vào trường. Không thay đổi bộ đệm.

-4: độ dài của chuỗi đầu vào vượt quá mức tối đa được chỉ định. Bộ đệm chứa chuỗi đầu vào tối đa cho phép cộng với giá trị rỗng kết thúc.

Thí dụ:

```
.dữ liệu
Message: .asciiz "Ho va ten sinh vien:"
string: .space 100
.chữ
    li $ v0, 54
    la $ a0, Tin nhắn
    la $ a1, chuỗi
    la $ a2, 100
    syscall
```

và kết quả là

9. ký tự in

In một ký tự ra đầu ra tiêu chuẩn (bảng điều khiển).

Tranh luận):

$ v0 = 11

$ a0 = ký tự cần in (ở byte quan trọng thấp nhất)

Giá trị trả lại:

không ai

Thí dụ:

```
    li $ v0, 11
    li $ a0, 'k'
    syscall
```

và kết quả là

10. đọc ký tự

Nhận một ký tự từ đầu ra tiêu chuẩn (bàn phím).

Tranh luận):

$ v0 = 12

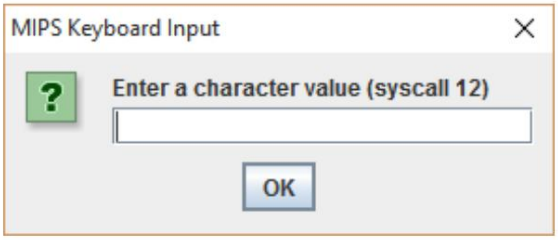
Giá trị trả lại:

$ v0 chứa ký tự đã đọc

Thí dụ:

```
    li $ v0, 12
    syscall
```

và kết quả là



11. ConfirmDialog

Hiển thị câu hỏi tin nhắn với 3 nút: Có | Không | Hủy bỏ

Tranh luận):

$ v0 = 50

$ a0 = địa chỉ của chuỗi thông báo kết thúc bằng rỗng

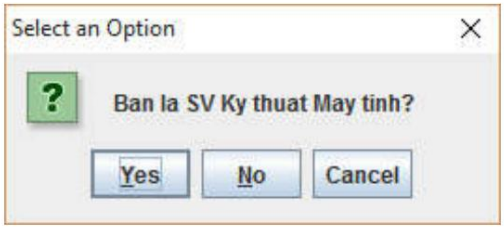
Giá trị trả lại:

$ a0 = chứa giá trị của tùy chọn do người dùng chọn
0: Có
1: Không
2: Hủy bỏ

Thí dụ:

```
.dữ liệu
Message: .asciiz "Ban la SV Ky thuat May tinh?"
.chữ
    li $ v0, 50
    la $ a0, Tin nhắn
    syscall
```

và kết quả là



12. MessageDialog

Chỉ hiển thị thông báo tin nhắn với biểu tượng và nút OK

Tranh luận):

$ v0 = 55

$ a0 = địa chỉ của chuỗi thông báo kết thúc bằng rỗng

$ a1 = loại thông báo sẽ được hiển thị:

0: thông báo lỗi, được biểu thị bằng biểu tượng Lỗi

1: thông báo thông tin, được biểu thị bằng biểu tượng Thông tin

2: thông báo cảnh báo, được biểu thị bằng biểu tượng Cảnh báo

3: thông báo câu hỏi, được biểu thị bằng biểu tượng Câu hỏi

khác: tin nhắn đơn giản (không có biểu tượng hiển thị)

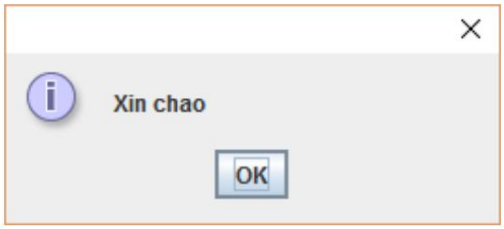
Giá trị trả lại:

không ai

Thí dụ:

```
.dữ liệu
Tin nhắn: .asciiz "Xin chao"
.chữ
    li $ v0, 55
    la $ a0, Tin nhắn
    syscall
```

và kết quả là



13. MIDIout

Tạo ra âm thanh

Tranh luận):

$ v0 = 31

$ a0 = sân (0-127)

$ a1 = thời lượng tính bằng mili giây

$ a2 = công cụ (0-127)

$ a3 = khối lượng (0-127)

Giá trị trả lại:

Tạo âm báo và trở lại ngay lập tức

Thí dụ:

```
    li $ v0, 33 li
    $ a0, 42 #pitch
    li $ a1, 2000 #time
    li $ a2, 0 #musical hướng dẫn
    li $ a3, 212 #volume
```

14. MIDIouts không đồng bộ

Tạo ra âm thanh

Tranh luận):

$ v0 = 33

$ a0 = sân (0-127)

$ a1 = thời lượng tính bằng mili giây

$ a2 = công cụ (0-127)

$ a3 = khối lượng (0-127)

Giá trị trả lại:

Tạo âm sắc và trở lại sau khi hoàn thành giai điệu

Thí dụ:

```
    li $ v0, 33 li
    $ a0, 42 #pitch
    li $ a1, 2000 #time
    li $ a2, 0 #musical hướng dẫn
    li $ a3, 212 #volume
    syscall
```

15. Lối ra

Đã chấm dứt phần mềm. Hãy hiểu rằng không có lệnh EXIT trong Bộ hướng dẫn của bất kỳ bộ xử lý nào. Thoát là một dịch vụ thuộc Hệ điều hành.

Tranh luận):

$ v0 = 10

Giá trị trả lại:

không ai

Thí dụ:

```
    li $ v0, 10 #lối ra
    syscall
```

16. Exitwithcode

Đã chấm dứt phần mềm. Hãy hiểu rằng không có lệnh EXIT trong Bộ hướng dẫn của bất kỳ bộ xử lý nào. Thoát là một dịch vụ thuộc Hệ điều hành.

Tranh luận):

$ v0 = 17

$ a0 = kết quả chấm dứt

Giá trị trả lại:

không ai

Thí dụ:

```
    li $ v0, 17 # thoát
    li $ a0, 3 # với mã lỗi = 3
    syscall
```

Bài tập tại nhà và tại phòng thí nghiệm

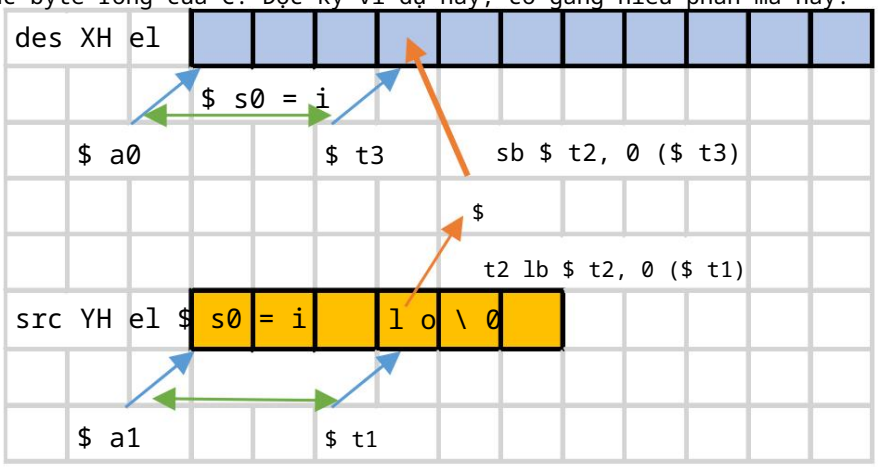
HomeAssignment1

Chương trình hợp ngữ đơn giản sau sẽ hiển thị một chuỗi chào mừng. Chúng tôi sử dụng hàm printf cho mục đích này. Đọc kỹ ví dụ này, chú ý cách truyền tham số cho hàm printf. Đọc Tham khảo Môi trường Phòng thí nghiệm Mips để biết thêm chi tiết.

```
# Bài tập Phòng thí nghiệm 5, Bài tập về nhà 1
 .dữ liệu
kiểm tra: .asciiz "Hello World"
 .chữ
    li $ v0, 4
    la $ a0, thử nghiệm
    syscall
```

HomeAssignment2

Thủ tục strcpy sao chép chuỗi y sang chuỗi x bằng cách sử dụng quy ước kết thúc byte rỗng của C. Đọc kỹ ví dụ này, cố gắng hiểu phần mã này.




```
# Bài tập Phòng thí nghiệm 5, Bài tập về nhà 2
.dữ liệu
x: .space 32                                    # chuỗi đích x, trống
y: .asciiz "Xin chào"                           # chuỗi nguồn y

.chữ
strcpy:
        thêm $ s0, $ 0, $ 0                     # $ s0 = i = 0
L1:
        thêm $ t1, $ s0, $ a1                   # $ t1 = $ s0 + $ a1 = i + y [0]
                                                # = địa chỉ của y [i]
        t1) lb thêm t2$ 0t3$ $                  # $ t2 = giá trị tại $ t1 = y [i]
        s0, $ a0                                # $ t3 = $ s0 + $ a0 = i + x [0] # = địa
                                                chỉ của x [i]
        sb      $ t2,0 ($ t3)                   # x [i] = $ t2 = y [i]
        beq $ t2, $ 0, end_of_strcpy # if y [i] == 0, thoát
        nop
        ghiền $ s0, $ s0,1 j                    # $ s0 = $ s0 + 1 <-> i = i + 1
                L1                              # ký tự tiếp theo
        nop
end_of_strcpy:
```

HomeAssignment3

Chương trình sau đây đếm độ dài của một chuỗi kết thúc bằng null. Đọc kỹ ví dụ này, phân tích từng dòng mã.

```
# Bài tập Phòng thí nghiệm 5, Bài tập về nhà 3
.dữ liệu
string:             .space 50                   "
Message1:           .asciiz "Nhap xau: .asciiz"
Message2: .text     Do dai xau la: "

chính:
get_string: # TODO

get_length: la $ a0, thêm chuỗi $ t0, $                     # $ a0 = địa chỉ (chuỗi [0])
                    không, $ zero check_char: thêm          # $ t0 = i = 0
$ t1, $ a0, $ t0                                            # $ t1 = $ a0 + $ t0
                                                            # = địa chỉ (chuỗi [i])
```

```
                    lb $ t2, 0 ($ t1) # $ t2 = string [i]
                    beq $ t2, $ 0, end_of_str # có phải là ký tự rỗng
                    không? add $ t0, $ t0, 1 # $ t0 = $ t0 + 1 -> i = i + 1
                    j check_char
end_of_str:
end_of_get_length:
print_length: # TODO
```

Bài tập1

Tạo một dự án mới để thực hiện chương trình trong Bài tập ở nhà 1. Biên dịch và tải lên trình mô phỏng. Chạy và quan sát kết quả. Chuyển đến phần bộ nhớ dữ liệu, kiểm tra cách chuỗi thử nghiệm được lưu trữ và đóng gói trong bộ nhớ.

Chuyển nhượng 2

Tạo một dự án mới để in tổng của hai thanh ghi $ s0 và $ s1 theo định dạng sau:

"Tổng của (s0) và (s1) là (kết quả)"

Bài tập3

Tạo một dự án mới để triển khai chương trình trong Home Assignment 2. Thêm các hướng dẫn khác để gán một chuỗi kiểm tra cho biến y và triển khai strcpy
hàm số. Biên dịch và tải lên trình mô phỏng. Chạy và quan sát kết quả.

Bài tập4

Hoàn thành Bài tập về nhà 3 với chức năng gọi điện để lấy một chuỗi từ hộp thoại và hiển thị độ dài cho hộp thoại tin nhắn.

Nhiệm vụ5

Viết chương trình cho phép người dùng nhập một chuỗi bằng cách nhập các chữ cái riêng lẻ. Quá trình nhập liệu sẽ bị chấm dứt khi người dùng nhấn Enter hoặc khi đó độ dài của chuỗi vượt quá 20 ký tự. In chuỗi ngược lại.

Kết luận

Trước khi bạn vượt qua bài tập trong phòng thí nghiệm, hãy nghĩ về những câu hỏi dưới đây:

- Sự khác biệt giữa chuỗi trong C và Java là gì?
- Trong C, với 8 byte, chúng ta có thể lưu trữ bao nhiêu ký tự?
- Trong Java, với 8 byte, chúng ta có thể lưu trữ bao nhiêu ký tự?