**BÁO CÁO THỰC HÀNH GIỮA KÌ**

**IT3280 – THỰC HÀNH KIẾN TRÚC MÁY TÍNH**

Họ và tên: Trần Thu Phương

MSSV: 20235811

**Assignment A.3**

* Đề bài: Viết hàm kiểm tra một số có phải là số nguyên tố không. Sau đó nhập 2 số nguyên dương M và N từ bàn phím, in ra tất cả các số nguyên tố trong đoạn từ M đến N.
* Phân tích cách thực hiện
* Đầu vào: Nhập 2 số nguyên dương M và N (0<M<=N), nếu đầu vào không hợp lệ yêu cầu người dùng nhập lại
* Khi đầu vào hợp lệ, duyệt từng phần tử từ M đến N để kiểm tra số nào là số nguyên tố
* Chương trình con kiểm tra tham số đầu vào t4 có là số nguyên tố không:
* Nếu t4 < 2 thì t4 không là số nguyên tố, gán s1=0
* Nếu t4>=2 thì ta dùng vòng lặp for(int i=2, i<=t4/2, i++) xem t4 có chia hết số i nào không, nếu có thì kết luận t4 không là số nguyên tố, ngược lại t4 là số nguyên tố, gán s1=1, dùng biến t0 để đếm số nguyên tố trong đoạn từ M đến N
* Kết quả trả về của chương trình con là s1=0 (t4 không là số nguyên tố) hoặc s1=1 (t4 là số nguyên tố)
* In ra các số nguyên tố trong đoạn từ M đến N
* Nếu t0=0 thì không có số nguyên tố nào trong đoạn M đến N
* Ngược lại thì in ra các số nguyên tố
* Kết quả thực hiện

TH1: Nhập đến khi kết quả đầu vào hợp lệ và có số nguyên tố trong đoạn M đến N

A white text with black text

Description automatically generated

TH2: Không có số nguyên tố nào trong đoạn từ M đến N

A black text on a white background

Description automatically generated

* Mã nguồn:

.data

nhapm: .asciz "Nhap so nguyen M: "

nhapn: .asciz "Nhap so nguyen N: "

ketqua: .asciz "Cac so nguyen to trong doan tu M den N: "

ketqua1: .asciz "Khong co so nguyen to nao trong doan tu M den N."

loi1: .asciz "Vui long nhap so nguyen duong.\n"

loi2: .asciz "Vui long nhap lai, M khong duoc lon hon N.\n"

daucach: .asciz " "

.text

# Nhập đầu vào

nhapM:

li a7, 4

la a0, nhapm

ecall

li a7, 5

ecall

bge zero, a0, nhaplaim # nếu M < 0 thì nhập lại

mv s2, a0 # M

nhapN:

li a7, 4

la a0, nhapn

ecall

li a7, 5

ecall

bge zero, a0, nhaplain # nếu N<0 thì nhập lại

mv s3, a0 # N

blt s3, s2, nhaplai # nếu M>N thì nhập lại

j hople

#khi đầu vào không hợp lệ

nhaplaim:

li a7, 4

la a0, loi1

ecall

j nhapM

nhaplain:

li a7, 4

la a0, loi1

ecall

j nhapN

nhaplai:

li a7, 4

la a0, loi2

ecall

j nhapN

#khi đầu vào hợp lệ

hople:

li a7, 4

la a0, ketqua

ecall

addi s3, s3, 1 # N+1

j loop

loop:

beq s2, s3, exits # i=M, nếu i=N+1 thì kết thúc vòng lặp

add t4, s2, zero # gán t4=M

jal kiemtrasnt # bắt đầu vào chương trình con để kiểm tra số nguyên tố

bne s1, zero, print # là số nguyên tố thì print

continue:

addi s2, s2, 1 #i++

j loop

# Hàm kiểm tra số nguyên tố

kiemtrasnt:

li t3, 2 # gán 2 vào t3 để tính t4/2

blt t4, t3, exit1 # t4<2 thì t4 không là số nguyên tố

div t5, t4, t3

addi t5, t5, 1 # t5=t4/2 + 1

li t6, 2 # j=2

loopsonguyento:

beq t6, t5, thoat # j=t5 thì kết thúc vòng lặp kiểm tra số nguyên tố

rem s0, t4, t6 #s0=t4%j

beq s0, zero, exit1 # nếu s0=0 thì t4 không là số nguyên tố

addi t6, t6, 1 # j++

j loopsonguyento

thoat:

li s1, 1 # t4 là số nguyên tố

addi t0, t0, 1 # t0 đếm số lượng số nguyên tố

j exit2

exit1:

li s1, 0 # t4 không là số nguyên tố

exit2:

jr ra # nhảy đến caller

exits:

beq t0, zero, tiep # nếu không có số nguyên tố nào trong đoạn từ M đến N

j ketthuc

tiep:

li a7, 4

la a0, ketqua1 # in thông điệp "Khong co so nguyen to nao trong doan tu M den N."

ecall

ketthuc:

li a7, 10 #exit

ecall

# in kết quả

print:

li a7, 1

mv a0, s2 # in số nguyên tố

ecall

li a7, 4

la a0, daucach # in dấu cách

ecall

j continue

**Assignment B.13**

* Đề bài: Nhập mảng số nguyên từ bàn phím. In ra màn hình số các phần tử có giá trị khác nhau trong mảng.
* Phân tích cách thực hiện:
* Đầu vào: Nhập số phần tử của mảng (n>=0), nếu không hợp lệ thì nhập lại. Nhập các phần tử trong mảng
* Sắp xếp mảng tăng dần (Bubble Sort):
* Duyệt từ đầu mảng đến cuối mảng, nếu array[j]>array[j+1] thì đổi chỗ, như vậy phần tử lớn nhất trong mảng sẽ về cuối mảng
* Tiếp tục duyệt như thế với mảng nhỏ hơn một phần tử cho đến khi còn một phần tử
* Đếm số phần tử khác nhau: Bắt đầu gán biến đếm s6=1, so sánh array[i] với array[i-1], nếu giống nhau thì không tăng đếm, khác nhau thì s6++
* Nếu n=0 thì mảng không có phần tử nào, nếu n>0 thì in ra biến đếm s6
* Kết quả thực hiện:

TH1: n=0

A close up of black text

Description automatically generated

TH2: n>0

A white background with black text

Description automatically generated

* Mã nguồn:

.data

array: .space 400 # Cấp phát 400 byte để lưu mảng (tối đa 100 phần tử số nguyên 4 byte)

msg1: .asciz "Nhap do dai mang n: "

msg2: .asciz "Nhap mang: \n"

msg3: .asciz " so khac nhau trong mang."

msg4: .asciz "Vui long nhap lai n (n>=0).\n"

msg5: .asciz "Mang khong co phan tu nao."

.text

nhapn: # Nhập số phần tử của mảng

li a7, 4

la a0, msg1 # in thông báo "Nhap do dai mang n: "

ecall

li a7, 5

ecall

mv s0, a0 # s0 = n

blt s0, zero, nhaplai # n<0 thì nhập lại

j continue

nhaplai: # Nhập lại n

li a7, 4

la a0, msg4

ecall

j nhapn

continue:

beq s0, zero, thoat # n=0 thì nhảy đến thoát

la s1, array # địa chỉ mảng array

li t0, 0 # i=0

li a7, 4

la a0, msg2 # in thông báo "Nhap mang"

ecall

nhapmang: # Nhập các phần tử của mảng

bge t0, s0, nhapxong # i >= n thì nhập xong

li a7, 5 # nhập giá trị

ecall

sw a0, 0(s1) # lưu giá trị vào mảng

addi s1, s1, 4 # tăng địa chỉ tới phần tử tiếp theo

addi t0, t0, 1 # i++

j nhapmang

nhapxong:

# Sắp xếp mảng

li t1, 0 # i=0

loopsapxep:

la s1, array

addi t2, s0, -1 # n-1

bge t1, t2, done # i>=n-1 thì kết thúc sắp xếp

li t3, 0 #j

loopdoicho:

sub t4, t2, t1 # n-1-i

bge t3, t4, exit # nếu j>n-1-i thì kết thúc lặp

slli t5, t3, 2 # t5=j\*4

add t6, s1, t5 # địa chỉ của array[j]

lw s7, 0(t6) # array[j]

lw s8, 4(t6) # array[j+1]

ble s7, s8, khongdoicho # array[j]<array[j+1] thì không đổi chỗ

sw s8, 0(t6)# ngược lại thì đổi chỗ

sw s7, 4(t6)

khongdoicho:

addi t3, t3, 1 # j++

j loopdoicho

exit:

addi t1, t1, 1 # i++

j loopsapxep

done:

# đếm số phần tử khác nhau trong mảng

la t0, array # t0=địa chỉ mảng

li s6, 1 # số phần tử khác nhau

li t1, 1 #i=1

loopdem:

beq t1, s0, print # i==n thì print

slli t2, t1, 2 # t2=4\*i

add t3, t0, t2 # địa chỉ của array[i]

lw t4, 0(t3) #array[i]

addi t5, t3, -4 # địa chỉ của array[i-1]

lw t6, 0(t5) #array[i-1]

beq t4, t6, khongtang # nếu giống nhau thì không tăng

addi s6, s6, 1 # nếu khác thì tăng đếm

khongtang:

addi t1, t1, 1 # i++

j loopdem

print:

# in đếm

li a7, 1

add a0, zero, s6

ecall

# in dòng " so khac nhau trong mang."

li a7, 4

la a0, msg3

ecall

li a7, 10

ecall

# trường hợp n=0

thoat:

li a7, 4

la a0, msg5 # in thông báo "Mang khong co phan tu nao."

ecall

**Assignment C.2**

* Đề bài: Nhập vào xâu ký tự. In ra các ký tự khác nhau có trong xâu.
* Phân tích cách thực hiện:
* Đầu vào: Xâu rỗng hoặc xâu khác rỗng, có thể chứa dấu cách
* Duyệt từng kí tự trong xâu, nếu gặp kí tự đặc biệt (dấu cách, ‘\n’) thì bỏ qua
* Dùng một mảng dem để đánh dấu các kí tự xuất hiện, nếu kí tự đã xuất hiện thì bỏ qua
* In ra các ký tự khác nhau trong xâu
* Kết quả thực hiện:

TH1: Xâu rỗng



TH2: Xâu khác rỗng (chứa dấu cách)

A black text on a white background

Description automatically generated

* Mã nguồn:

.data

string: .space 400 # Vùng nhớ để lưu xâu người dùng nhập (tối đa 400 byte)

dem: .space 400 # Mảng để đánh dấu các ký tự đã xuất hiện

msg1: .asciz "Nhap xau: "

msg2: .asciz "Cac ki tu khac nhau trong xau la: "

msg3: .asciz "Xau rong."

daucach: .asciz " "

.text

# Nhập xâu

li a7, 4

la a0, msg1 # in thông báo "Nhap xau: "

ecall

li a7, 8

la a0, string

li a1, 400

ecall

# Kiểm tra xem xâu có rỗng không

la s0, string # s0 trỏ đến đầu xâu

lb t6, 0(a0) # Lấy ký tự đầu tiên của xâu

beq t6, zero, rong # Nếu là '\0' thì xâu rỗng

li x3, 10 # mã ASCII của '\n'

beq t6, x3, rong # Nếu là ký tự xuống dòng thì xâu rỗng

li a7, 4

la a0, msg2 # in thông báo "Cac ki tu khac nhau trong xau la: "

ecall

la s1, dem # địa chỉ mảng dem

loop:

lb s2, 0(s0) # s2 = string[i]

beq s2, zero, exit # Nếu là ký tự NULL thì thoát

li x2, 32 # mã ASCII của dấu cách

beq s2, x2, continue # Nếu là dấu cách thì bỏ qua

beq s2, x3, continue # Nếu là ký tự xuống dòng thì bỏ qua

add t5, s1, s2 # địa chỉ của dem[string[i]]

lb t3, 0(t5) # dem[string[i]]

bne t3, zero, continue # Nếu ký tự đã xuất hiện thì bỏ qua

li t4, 1 # Đánh dấu ký tự s2 đã xuất hiện

sb t4, 0(t5) # dem[string[i]]=1

# In ra kí tự khác nhau

li a7, 11

mv a0, s2

ecall

li a7, 4

la a0, daucach # in dấu cách

ecall

continue:

addi s0, s0, 1 # i++

j loop

exit:

li a7, 10 # exit

ecall

# Nếu xâu rỗng

rong:

li a7, 4

la a0, msg3

ecall

li a7, 10

ecall