Báo cáo giữa kỳ

Phùng Ngọc Vinh – 20194719

Bài A3:

1.Cách thực hiện:

Để nhận biết được số M là số nguyên tố: ta cho dùng vòng lặp từ 2 → M/2. Nếu số M chia hết cho một số bất kỳ trong khoảng (2;M/2) thì số đó không là số nguyên tố.

Để in dãy số nguyên tố nhỏ hơn M: ta dùng vòng lặp từ (0;M) và sử dụng điều kiện nhận biết ở trên để lưu các số nguyên tố vào mảng.

2.

Nhập số nguyên M từ bàn phím:

```
nhap_M:                         # nhap so nguyen M
        addi  $v0,$0, 51
        la      $a0, Message_M
        syscall
        addi  $s1, $a0, 0 # $s1 = $a0 = M
end_nhap:
```

Chương trình con kiểm tra có là số nguyên tố:

```
isPrime:
dk1:  slti    $t2,$t0,2    # $t2 = 1 neu t0 < 2
        beq   $t2, $0, dk2        # t2 = 0 thi branch dk2
        addi  $v0, $0, 1   # v0 = 1 (t0 khong phai so nguyen to)
        j       end_isPrime
```

```
end_dk1:
dk2:  addi  $s3,$0,2          #i = 2 = s3
      bne   $t0, $s3, else    # neu t0 != 2 thi branch else
      add   $v0,$0,$0  # v0 = 0 (t0 la so nguyen to)
      j     end_isPrime
end_dk2:
else: div   $s4,$t0,$s3             # s4  = t0/2
      addi  $v0, $0,0         # t3 = 0
loop2:      blt   $s4,$s3, end_loop2    #neu s4 < s3 (t0/2 < i) thi branch
end_for
      div   $t0,$s3                 # t0 chia s3
      mfhi  $t4               # t4 = t0 mod s3 (lay gia tri $hi vao $t4)
      addi  $s3,$s3,1         # i = i+1
      beq   $t4,$0, notPrime      # neu t4 = 0 thi branch notPrime
      j     loop2
notPrime:  addi  $v0,$v0,1  # t3 = 1 (t0 khong phai la so nguyen to)
end_loop2:
end_else:
end_isPrime:    jr    $ra
```

In kết quả:

* nếu không tìm thấy số nguyên tố thỏa mãn điều kiện nhỏ hơn M:

```
null:  bne   $t7,$0, printf     # t7 != 0 branch printf
```

```
            li      $v0, 55
            la      $a0, Null
            li      $a1, 1
            syscall                 # khong tim thay so nguyen to nao
endNull:    j       exit
```

* Nếu tìm thấy số nguyên tố thỏa mãn điều kiện nhỏ hơn M:

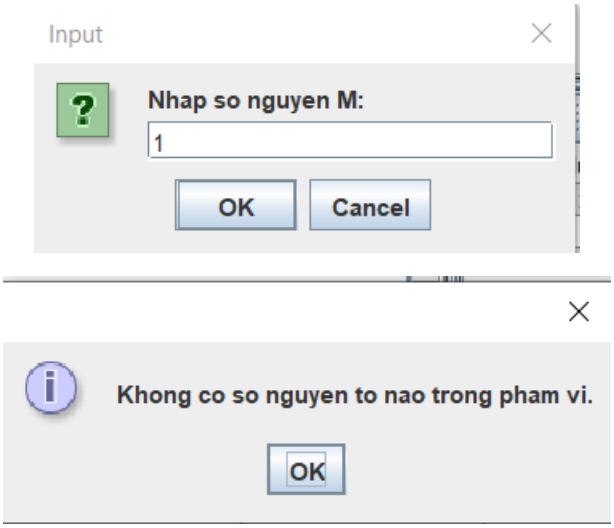
```
printf:
        li      $v0, 4
        la      $a0, Mess
        syscall


        li      $t8, 0                  #j = 0
printPrime:
        beq   $t7, $t8, end_printPrime
        sll     $t9, $t8, 2  # 4j
        add   $s7, $s2,$t9       # address A[j]
        lw      $s6,0($s7)  # s6 = A[j]
        li $v0, 1               # print A[i]
        add $a0,$0,$s6
        syscall
        addi $v0, $0, 4             # print ' '
        la      $a0, space
```

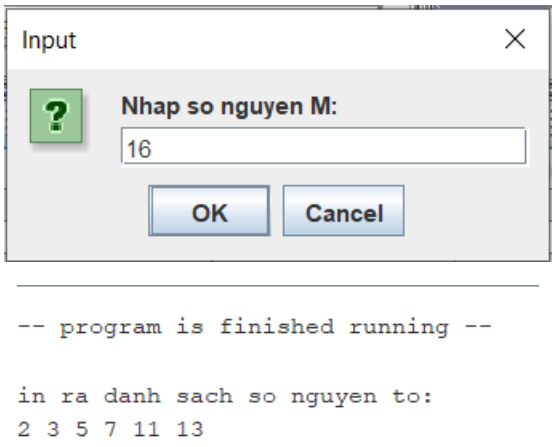
```
        syscall
        addi $t8, $t8 ,1          # j = j+1
        j       printPrime
end_printPrime:
end_printf:       j       exit
```

3.Kết quả chạy:

M = 1:



M = 16:



4.Mã nguồn:

```
.data
A:      .word
Message_M:      .asciiz "Nhap so nguyen M:"
Null:           .asciiz "Khong co so nguyen to nao trong pham vi."
space:          .asciiz " "
Mess:           .asciiz  "in ra danh sach so nguyen to:\n"
.text
main:

nhap_M:                         # nhap so nguyen M
        addi  $v0,$0, 51
        la      $a0, Message_M
        syscall
        addi  $s1, $a0, 0 # $s1 = $a0 = M
end_nhap:
        la      $s2,A         # lay dia chi co so mang A vao $s2
        addi  $t7,$0,0     #index of array A

        addi  $t0,$0,0     #khoi tao t0 = N
loop1:      slt     $t1,$s1, $t0              # t1 = 1 neu M < t0
        bne   $t1,$0, end_loop1      # neu t1 = 1 thi branch end
        jal     isPrime             # goi thu tuc isPrime
```

```
        bne   $v0,$0, tang_dem      #neu t3 = 1 thi branch tang_dem
        sll   $t6,$t7,2          #t6 =4 * t7 = 4 * index
        add   $t6,$t6,$s2             # t6 = address A[index]
        addi  $t7,$t7,1          # index = index+1
        sw    $t0,0($t6)         #A[index] = t0;
tang_dem: addi $t0,$t0,1             # t0 = t0+1
        j     loop1
end_loop1:
null:  bne   $t7,$0, printf     # t7 != 0 branch printf
               li     $v0, 55
               la     $a0, Null
               li     $a1, 1
               syscall                  # khong tim thay so nguyen to nao
endNull:    j     exit

printf:
        li     $v0, 4
        la     $a0, Mess
        syscall

        li     $t8, 0                   #j = 0
printPrime:
        beq   $t7, $t8, end_printPrime
```

```
        sll     $t9, $t8, 2 # 4j
        add   $s7, $s2,$t9       # address A[j]
        lw     $s6,0($s7) # s6 = A[j]
        li $v0, 1              # print A[i]
        add $a0,$0,$s6
        syscall
        addi $v0, $0, 4           # print ' '
        la      $a0, space
        syscall
        addi $t8, $t8 ,1          # j = j+1
        j       printPrime
end_printPrime:
end_printf:        j       exit

exit:  addi  $v0,$0, 10 #exit
        syscall

end_main:

isPrime:
dk1:  slti    $t2,$t0,2   # $t2 = 1 neu t0 < 2
        beq   $t2, $0, dk2       # t2 = 0 thi branch dk2
        addi  $v0, $0, 1  # v0 = 1 (t0 khong phai so nguyen to)
```

```
        j       end_isPrime
end_dk1:
dk2:  addi  $s3,$0,2            #i = 2 = s3
        bne   $t0, $s3, else      # neu t0 != 2 thi branch else
        add   $v0,$0,$0  # v0 = 0 (t0 la so nguyen to)
        j       end_isPrime
end_dk2:
else:  div    $s4,$t0,$s3               # s4  = t0/2
        addi  $v0, $0,0           # t3 = 0
loop2:        blt     $s4,$s3, end_loop2     #neu s4 < s3 (t0/2 < i) thi branch
end_for
        div    $t0,$s3                  # t0 chia s3
        mfhi  $t4                  # t4 = t0 mod s3 (lay gia tri $hi vao $t4)
        addi  $s3,$s3,1           # i = i+1
        beq   $t4,$0, notPrime        # neu t4 = 0 thi branch notPrime
        j       loop2
notPrime:  addi  $v0,$v0,1  # t3 = 1 (t0 khong phai la so nguyen to)
end_loop2:
end_else:
end_isPrime:      jr      $ra
```

Bài B3:

1. Cách thực hiện
2. D
3. Chạy kết quả

   Đầu vào: Array[] = { -1, 150, 190, 170, -1, -1, 160, 180}

```
Nhap gia tri phan tu mang !
Mang truoc khi sap xep:
 -1 150 190 170 -1 -1 160 180
Mang sau khi sap xep:
 -1 150 160 170 -1 -1 180 190
-- program is finished running --
```

Bài C1:

1.Cách thực hiện:

Ta nhập vào mảng và độ dài của mảng.

Kiểm tra nếu độ dài mảng bằng 1 thì in ra kết quả xâu nhập vào là xâu đối xứng.

```
slti   $t0, $a0, 2
bne    $t0, $zero, returnTrue
```

Sau đó so sánh kí tự đầu tiên và kí tự cuối cùng của mảng.

Nếu không giống nhau thì trả về kết quả xâu nhập vào không là xâu đối xứng.

```
lb     $t0, 0($a1)
addi   $t1, $a0, -1
add    $t1, $t1, $a1
lb     $t1, 0($t1)
bne    $t0, $t1, returnFalse
```

Sau đó thay đổi độ dài xâu (giảm 2 ký tự) và thay đổi địa chỉ con trỏ (trỏ vào kí tự tiếp theo) và quay lại vòng lặp

```
addi  $a0, $a0, -2
addi  $a1, $a1, 1
j     ktradoixung
```

2.

Nhập đầu vào:

```
input: .asciiz     "abc121cba5"
input_len: .word 10
```

Kiểm tra đối xứng:

```
ktradoixung:
    # kiem tra truong hop
    slti   $t0, $a0, 2
    bne    $t0, $0, returnTrue

    # so sanh ky tu dau va cuoi
    lb     $t0, 0($a1)
    addi   $t1, $a0, -1
    add    $t1, $t1, $a1
    lb     $t1, 0($t1)
    bne    $t0, $t1, returnFalse
```

```
# thay doi con tro, do dai va quay lai vong lap
addi  $a0, $a0, -2
addi  $a1, $a1, 1
j       ktradoixung
```

3.Kết quả chạy

Xâu: “Phung Ngoc Vinh”

Độ dài: 15

```
-- program is finished running --

Phung Ngoc Vinh la khong xau doi xung!
```

Xâu: “abc121cba”

Độ dài: 9

```
abc121cba la xau doi xung!
-- program is finished running --
```

Xâu: “abc121cba”

Độ dài: 12

```
abc121cba la khong xau doi xung!
-- program is finished running --
```

(Giải thích:

Do nhập vào độ dài xâu không đúng nên máy sẽ đọc vào cả ký tự Null cho đủ độ dài 12.

Dẫn đến kết quả như trên)