



Assembler - Flow Control

Demonstrate the ability to write assembly code with flow control.

Rules

Read carefully and check all rules you agree with:

- ☐ Each card has description which must be strictly followed.
- ☐ Keep the code properly formatted (correct indentation, line width is 40 characters max).
- ☐ Your code must represent your own individual work. If something is not clear, ask your instructor for help.
- ☐ Cheating of any kind (copying someone else's work, allowing others to copy your work, collaborating, etc.) will not be tolerated and will be dealt with SEVERELY.

Problem

- Given two circles with integral centers (x_1, y_1) , (x_2, y_2) , and radii r_1 , r_2 .
- Create the code that determines intersection or non-intersection of the circles.
- Your code must output "intersecting" or "non-intersecting" based on the result of calculation.
- Place your the same code in the middle of the following test cases. Do not modify given values in the tests.
- Use the most efficient method (without square root calculations)



Test 1

1



Test 2

1



Run

```
.data
x1:    .long    -5
y1:    .long    -6
r1:    .long    20
x2:    .long    7
y2:    .long    8
r2:    .long    30
str1:  .asciz   "intersecting"
str2:  .asciz   "non-intersecting"
.text
.global main
main:
    push    %rbx # For alignment

    mov x2, %eax
    sub x1, %eax
    imul %eax, %eax

    mov y2, %ecx
    sub y1, %ecx
    imul %ecx, %ecx

    add %eax, %ecx

    mov r1, %edx
    add r2, %edx
    imul %edx, %edx

    CMP %ecx, %edx
    JG nonintersect

    mov $str2, %rdi
    call printf
    jmp end

nonintersect:
    mov $str1, %rdi
    call printf

end:
    xor     %eax, %eax # return 0;
    pop     %rbx
    ret
```

intersecting

Run

```
.data
x1:    .long    -10
y1:    .long    -20
r1:    .long    2
x2:    .long    30
y2:    .long    40
r2:    .long    4
str1:  .asciz   "intersecting"
str2:  .asciz   "non-intersecting"
.text
.global main
main:
    push    %rbx # For alignment

    mov x2, %eax
    sub x1, %eax
    imul %eax, %eax

    mov y2, %ecx
    sub y1, %ecx
    imul %ecx, %ecx

    add %eax, %ecx

    mov r1, %edx
    add r2, %edx
    imul %edx, %edx

    CMP %ecx, %edx
    JG nonintersect

    mov $str2, %rdi
    call printf
    jmp end

nonintersect:
    mov $str1, %rdi
    call printf

end:
    xor     %eax, %eax # return 0;
    pop     %rbx
    ret
```

non-intersecting

Survey

- What is approximate number of hours you spent implementing this assignment?

1hr

- Indicate the specific portions of the assignment that gave you the most trouble

Test 2

By signing this document you fully agree that all information provided therein is complete and true in all respects.

Responder sign: