There are several symbolic machine languages for Intel-based processors. This example has been made using GNU Assembly language. (Another commonly used such language is NASM.)

There are a couple of lines if code (in the beginning and in the end) in order to make the program elf-compatible.

As a whole, the program code for symbolic machine language program nj.s is the following:

```
        .text
        .global main
main:
        # for ELF compatibility, start
        pushq   %rbp
        movq    %rsp, %rbp
        # for ELF compatibility, end
_L0:
        # printing x
        movq    $0, %rsi
        movq    $message, %rdi
        movq    $3, %rax
        call    printf
        # calculating a new value for x
        movsd   N, %xmm0
        divsd   x, %xmm0
        addsd   x, %xmm0
        mulsd   p, %xmm0
        movsd   %xmm0, x
        # printing value of x
        movq    %xmm0, %rax
        movq    %rax, %rsi
        movq    $format, %rdi
        movq    $12, %rax
        call    printf
        # iterations left?
        movq    i, %rax
        subq    $1, %rax
        movq    %rax, i
        cmpq    $0, i
        jnz     _L0
        # for ELF compatibility, start
        leave
        ret     $0
        # for ELF compatibility, end
        .data
format: .string "%12.6f\n"
message:.string "x: "
N:      .double 7.0
x:      .double 67.0
p:      .double 0.5
i:      .quad 10
```

The program can be compiled to an executable code in Linux environment by using gcc compiler – for example like this:

$ gcc -o prog nj.s

This creates a compiled program named ”prog” from the symbolic machine language program named “nj.s”. The program can then be executed by command

$ ./prog

Depending on the environment and compiler assumptions, additional attributes may be needed for the compiler – for example -no-pie. In this case the compiling command is

$ gcc -no-pie -o prog nj.s

Compile and execute the program. Test it and find out what it calculates, before you move on to modifying it and answering the actual questions in task 4.