Middle East Technical University
Department of Computer Engineering

CENG 140

Spring 2020

Take Home Exam 1

REGULATIONS

Due date: 22 March 2020, Sunday (by 23:59) *(Not subject to postpone)*

Submission: Electronically. You will be submitting your program source code written in a file which you will name as `the1.c` through the **cengclass** system. Resubmission is allowed (till the last moment of the due date), The last will replace the previous.

Team: There is **no** teaming up. The take home exam has to be done/turned in individually.

Cheating: **This is an exam:** all parts involved (source(s) and receiver(s)) get zero+parts may be subject to disciplinary action.

PROBLEM

This take home exam is about nested loop generation at run time. At run time you will be reading lines each of which define a loop variable (which is a single letter), a start value integer, an end value integer and a step size integer. The first line is the outermost loop and the last line is the inner most.

The nested loops will function as you would expect and after every innermost loop change a function with the name `loop_execute()` will be called. `loop_execute()` is a function that will be provided at evaluation-time by us (the evaluators). So, you will let the compiler know about it by including a header file we provide as addendum to this the description.

Here is a possible input (a very simple one):

```
X 10 18 3
r -3 2 1
Y 0 -2 -1
```

The outermost loop is defined in the first line. `X` will go through the values `10,13,16`. For each value of `X`, `r` will go through the values `-3,-2,-1,0,1,2`. The innermost loop is the last line. `Y` will go through the values `0,-1,-2`.

So, unless an alternation takes place (what this means will be explained below), `loop_execute()` will be called $3 \times 6 \times 3 = 54$ times.

`loop_execute()` will have access to the values of the so called loop variables. For this purpose a function that you will name as `loop_variable_value()` and implement will serve. For example `loop_variable_value('r')` will return the current value of the loop variable 'r' (considering the example given above).

Now comes a tough part: It will also be possible to continue with the next value of any loop variable. In that case, as expected, all inner loops will be reset to start. For this purpose a function that you will name as `loop_continue()` and implement will serve. Similar to `loop_variable_value()`, `loop_continue()` will also take a variable letter as argument. So, if the variables are at `X:10, r:-2, Y:-1` and inside `loop_execute()` a call to `loop_continue('r')` is performed then the next `loop_execute()` will see `X:10, r:-1, Y:0`.

SPECIFICATIONS

- The type of the numbers in the input complies with `int`. They will be separated by one or more blanks. There will be at least one line of loop definition.
- Do not even think about nesting $(2 \times \langle$letter count in alphabet$\rangle)$ real `for` loops in your program. Your program will receive a straight zero.
- It is possible that for a loop the given values are such that *start value* $>$ *end value* with a positive step size. (or the other way around for a negative step size). In this case neither that loop nor those that are 'inside' it will have a chance to be executed. Since `loop_execute()` is in the innermost loop, `loop_execute()` will not be executed even once (and the program will terminate).
- If `loop_execute()` contains `loop_continues()` (possibly at multiple positions) it is guarantied that no call to `loop_variable_value()` is performed after any execution of the `loop_continues()`.
- You can assume that the input is error free and complies with the input definition above.
- Since your programs are graded using some automated i/o DO NOT output ANYTHING.
- Your code shall start with

```
#include "loop.h"
```

The content of `loop.h` is as follows:

```
extern void loop_execute(void);
int loop_variable_value(char c);
void loop_continue(char c);
```

You are expected to implement `loop_variable_value()` and `loop_continue()` but not `loop_execute()`. In your code that you will submit, you shall make sure to call `loop_execute()`. In the evaluation process, your code will be compiled. `loop_execute()` will be coded by the evaluators and compiled separately. Then both will be linked together and run on test data.

Here is a `loop_execute()` definition which you can use to test your program for the example input above.
DO NOT FORGET TO REMOVE IT WHEN YOU SUBMIT.
ALSO DO NOT FORGET TO INCLUDE `loop.h` IN THE CODE YOU SUBMIT.

`loop_execute()` definition that you can use for testing.

```
void loop_execute(void)
{
 printf("X:%d r:%d Y:%d\n",
        loop_variable_value('X'),
        loop_variable_value('r'),
        loop_variable_value('Y'));
}
```

AN EXAMPLE RUN:

This is the output that would be generated with the example input and the example `loop_execute()` definition:

```
X:10 r:-3 Y:0
X:10 r:-3 Y:-1
X:10 r:-3 Y:-2
X:10 r:-2 Y:0
X:10 r:-2 Y:-1
X:10 r:-2 Y:-2
X:10 r:-1 Y:0
X:10 r:-1 Y:-1
X:10 r:-1 Y:-2
X:10 r:0 Y:0
X:10 r:0 Y:-1
X:10 r:0 Y:-2
X:10 r:1 Y:0
X:10 r:1 Y:-1
X:10 r:1 Y:-2
X:10 r:2 Y:0
X:10 r:2 Y:-1
X:10 r:2 Y:-2
X:13 r:-3 Y:0
X:13 r:-3 Y:-1
X:13 r:-3 Y:-2
X:13 r:-2 Y:0
X:13 r:-2 Y:-1
X:13 r:-2 Y:-2
X:13 r:-1 Y:0
X:13 r:-1 Y:-1
X:13 r:-1 Y:-2
X:13 r:0 Y:0
X:13 r:0 Y:-1
X:13 r:0 Y:-2
X:13 r:1 Y:0
X:13 r:1 Y:-1
X:13 r:1 Y:-2
X:13 r:2 Y:0
X:13 r:2 Y:-1
X:13 r:2 Y:-2
X:16 r:-3 Y:0
X:16 r:-3 Y:-1
X:16 r:-3 Y:-2
X:16 r:-2 Y:0
X:16 r:-2 Y:-1
X:16 r:-2 Y:-2
X:16 r:-1 Y:0
X:16 r:-1 Y:-1
X:16 r:-1 Y:-2
X:16 r:0 Y:0
X:16 r:0 Y:-1
X:16 r:0 Y:-2
X:16 r:1 Y:0
X:16 r:1 Y:-1
X:16 r:1 Y:-2
X:16 r:2 Y:0
X:16 r:2 Y:-1
X:16 r:2 Y:-2
```

Here is a another `loop_execute()` which makes use of `loop_continue()`:

```
void loop_execute(void)
{
    printf("X:%d r:%d Y:%d\n",
           loop_variable_value('X'),
           loop_variable_value('r'),
           loop_variable_value('Y'));
    if ((loop_variable_value('X')+loop_variable_value('r')+loop_variable_value('Y')) % 5 == 0)
      { printf("JUST MAGIC OF FIVE HAPPENED!...Continuing with next 'r' value.\n");
        loop_continue('r'); }
}
```

And here is the output:

```
X:10 r:-3 Y:0
X:10 r:-3 Y:-1
X:10 r:-3 Y:-2
JUST MAGIC OF FIVE HAPPENED!...Continuing with next 'r' value.
X:10 r:-2 Y:0
X:10 r:-2 Y:-1
X:10 r:-2 Y:-2
X:10 r:-1 Y:0
X:10 r:-1 Y:-1
X:10 r:-1 Y:-2
X:10 r:0 Y:0
JUST MAGIC OF FIVE HAPPENED!...Continuing with next 'r' value.
X:10 r:1 Y:0
X:10 r:1 Y:-1
JUST MAGIC OF FIVE HAPPENED!...Continuing with next 'r' value.
X:10 r:2 Y:0
X:10 r:2 Y:-1
X:10 r:2 Y:-2
JUST MAGIC OF FIVE HAPPENED!...Continuing with next 'r' value.
X:13 r:-3 Y:0
JUST MAGIC OF FIVE HAPPENED!...Continuing with next 'r' value.
X:13 r:-2 Y:0
X:13 r:-2 Y:-1
JUST MAGIC OF FIVE HAPPENED!...Continuing with next 'r' value.
X:13 r:-1 Y:0
X:13 r:-1 Y:-1
X:13 r:-1 Y:-2
JUST MAGIC OF FIVE HAPPENED!...Continuing with next 'r' value.
X:13 r:0 Y:0
X:13 r:0 Y:-1
X:13 r:0 Y:-2
X:13 r:1 Y:0
X:13 r:1 Y:-1
X:13 r:1 Y:-2
X:13 r:2 Y:0
JUST MAGIC OF FIVE HAPPENED!...Continuing with next 'r' value.
X:16 r:-3 Y:0
X:16 r:-3 Y:-1
X:16 r:-3 Y:-2
X:16 r:-2 Y:0
X:16 r:-2 Y:-1
X:16 r:-2 Y:-2
X:16 r:-1 Y:0
JUST MAGIC OF FIVE HAPPENED!...Continuing with next 'r' value.
X:16 r:0 Y:0
X:16 r:0 Y:-1
JUST MAGIC OF FIVE HAPPENED!...Continuing with next 'r' value.
X:16 r:1 Y:0
X:16 r:1 Y:-1
```

```
X:16 r:1 Y:-2
JUST MAGIC OF FIVE HAPPENED!...Continuing with next 'r' value.
X:16 r:2 Y:0
X:16 r:2 Y:-1
X:16 r:2 Y:-2
```