GIT CSE102 HW05

Fall 2014

Due Date

30.10.2014, 23:59

Implement the following functions;

1. **(20 pts.) Write the function** read_time that reads time-of-the-day information from user. The function should read the time in 24-hour format hh:mm where hh is hour and mm is minute. The function should return the hour and the minute in output parameters. The function should return **-1** as hour if the time is not valid. Use **do_while statement**s if you need to write a loop. Note that there could be several **space characters** before hour, between hour and ':' character and between ':' character and minute. The function should accept the following inputs as valid time:

```
15:15
   5 :20
14     :     30
```

The following inputs are not valid:

```
35:40
12:-12
12 30
15-20
```

Do not forget to skip the rest of the input line before leaving the function.

If the input is valid, print the resulting time in the format:

```
Hour:15
Minute:15
```

If the input is not valid, print;

```
Input is not valid.
```

2. **(20 pts.) Write the function** print_time that prints a time in a diamond shape. The function takes three input parameters; hour, minute and dimond_size. The time should be prınted at the center of the diamond shape. Use **for statements** if you need to write a loop. Note that the size of the diamond should be at least 4. If the diamond-size is less than 4 the function should print the time only without a diamond shape. If the diamond_size is 5 the function should output:

```
          *
         * *
        *   *
       *     *
      *       *
     *  12:50  *
      *       *
       *     *
        *   *
         * *
          *
```

If the diamond_size is 2 the function should output:

```
12:30
```

3. **(20 pts.) Write the function** increment_time that advances a time with a given integer. The function takes an integer input parameter, minutes, and increments the time (assume that the time is valid) passed as an input/output parameter (you should get time information from the user via read_time function, it means you should check input validity indirectly). For example, 10:40 incremented with 100 minutes is 12:20 and 23:30 incremented with 50 minutes is 00:20. Note that your function should also accept negative minute values. For example, 10:40 incremented with -110 minutes is 08:50 and 01:30 incremented with -120 minutes is 23:30. Finally, print the incremented time to screen.
4. **(20 pts.)** A function that manipulates input file according to its *function pointer* parameter. The function should print the manipulated file content to the console. You have to implement following function pointers;
   a. `void uppercase(char content[]); //converts all letters to uppercase`
   b. `void lowercase(char content[]); // converts all letters to lowercase`
   c. `void capitalize(char content[]); //capitalizes each word`
   d. `void toggle(char content[]); // converts lowercase letters to uppercase and uppercase letters to lowercase`

   Header:

```
void text_file_manipulator( char str_filename[],void
(*pf_convertion)( char content[]));
```

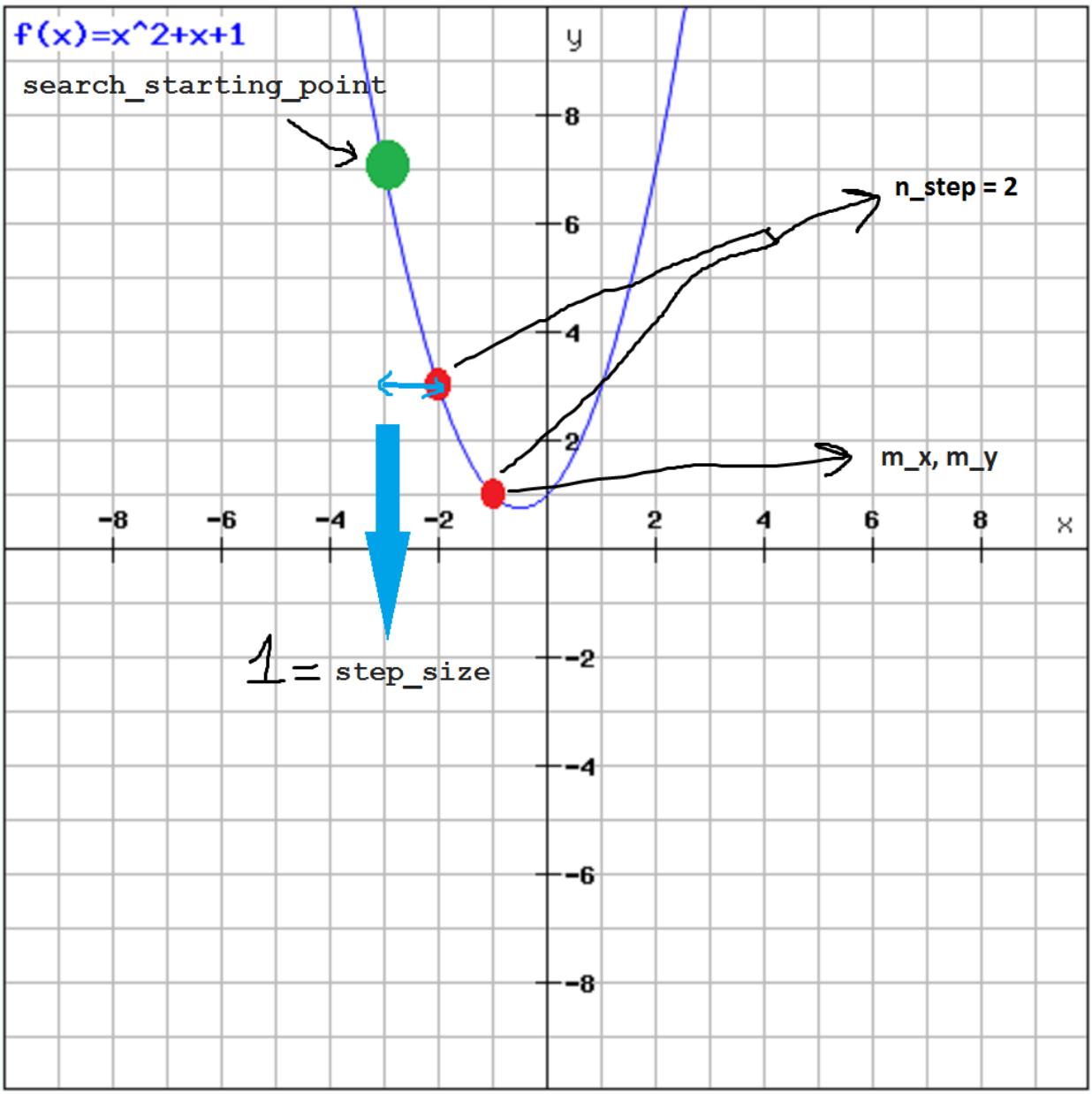
5. **(20 pts.)** A function that finds approximate maximum or minimum point of a second degree polynomial function (the point where the derivation will equal to zero). The input polynomial function will be in the following format: $x^2 + bx + c = 0$. Your C function should take a , b and c as input parameters. Your C function also should take the `search_starting_point` and `step_size` from the user. Finally, print the resulting maximum or minimum point (m_x, m_y) and step count (n_step) in your function.

   For example, if the input is (`a, b, c, search_starting_point, step_size`):
   1 1 1 -3 1
   Output would be similar to this;

```
Maximum point results ( m_x, m_y, n_step )
-1   1   2
```

   Header:

```
void max_finder( double a, double b, double c, double
search_starting_point, double step_size);
```


f(x)=x^2+x+1
search_starting_point
n_step = 2
m_x, m_y
1 = step_size
y
x
8
6
4
2
-2
-4
-6
-8
-8
-6
-4
-2
2
4
6
8

Your main function must be in following format;

```
int main(){
     //You have to correct all compile errors

     ////////////////////////////////////////
     puts("-------------------------------");
     printf("read_time:\n");
     read_time( hour, minute);
     puts("-------------------------------");
     ////////////////////////////////////////


     ////////////////////////////////////////
     puts("-------------------------------");
     printf("print_time:\n");
     scanf ("%d %d %d", &d_hour, &d_minute, &d_size);
     print_time ( d_hour, d_minute, d_size);
     puts("-------------------------------");
     ////////////////////////////////////////


     ////////////////////////////////////////
     puts("-------------------------------");
     printf("increment_time:\n");
     scanf ("%d ", &inc);
     increment_time (inc);
     puts("-------------------------------");
     ////////////////////////////////////////


     ////////////////////////////////////////
     puts("-------------------------------");
     printf("text_file_manipulator:\n");
     text_file_manipulator("in.txt", uppercase);
     text_file_manipulator("in.txt", lowercase);
     text_file_manipulator("in.txt", capitalize);
     text_file_manipulator("in.txt", toggle);
     puts("-------------------------------");
     ////////////////////////////////////////


     ////////////////////////////////////////
     puts("-------------------------------");
     printf("max_finder:\n");
     scanf ("%lf %lf %lf %lf %lf ", &a, &b, &c, &search_starting_point,
     &step_size);
     max_finder (a , b, c, search_starting_point, step_size);
     puts("-------------------------------");
     ////////////////////////////////////////



}
```

Notes:

- You should submit 1 files;
  - main.c
- Add all files into a folder and compress it for submission. The folder names will be restricted to the following format:
  HW#_studentid_studentname.
  - Example:
    HW01_121044001_Abdullah_Akay
- Upload soft copy of your homework to Moodle course web page
- **DON'T submit hard copy of your assignment**.
- Don't forget to test your code in the provided Linux virtual machine.
- Obey good programming rules (Indentation, Documenting, Well Commenting, Avoidıng magic numbers, Non-ascii characters etc.)
- **Strictly follow submission and file, folder naming rules.**