Object Oriented Programming

Fall 2020

Assignment 1

Seon Joo Kim
Yonsei University

Outline

- Deadline
- No Plagiarism
- Leave Comments
- Scoring
- Problem 1
- Problem 2
- Problem 3
- Submission
- Questions

Deadline

- Tuesday, 15th September 23:55
- No late submissions at all
- Normally, the deadline will be set on Monday starting from next assignment
- We are giving you a day more because it's the first assignment so please take a day to learn about file i/o and tar zip/unzip
  - it's described at the end of the slide (Appendix)
- The format of submission will be not be changed much, so taking time on the first assignment will ease your rest of the course

No Plagiarism

- No Mercy.

- The punishment will be made to both
  - the person who copied the code, and the person who shared the code.

Leave Comments

- Leave comments in your file for TAs to understand your code

- If no comments in the file, there may be a reduction of points

Scoring

- Problem1 (30%)
- Problem2 (35%)
- Problem3 (35%)

- If your code outputs correctly for given example input#.txt file
  - 30% base score per problem
- There will be additional 20 test cases
  - (70 / 20) = 3.5% per each case

- Ex) problem 1 all correct / problem 2 base score + 10 test case correct / problem 3 base score + 15 test case correct

  → 30 + 35*(0.3 + 0.035*10) + 35*(0.3 + 0.035*15) = 81.625

Scoring

- You should take care of your code not terminating by an issue in the middle of the loop
  - Scores will be given only by the final outputted file

- Example
  - 5 test cases
  - If your code is correct as O X O O O if ran seperately

  but terminates in the second test case by an error

  only the first test case is considered correct

Problem 1

- Find points where the gradient of the given cubic function is zero.

- The input will be interpreted as below

  $y = ax^3 + bx^2 + cx + d \qquad (a > 0)$

- Use *double* datatype for floating numbers
- Print numbers into precision of three

Problem 1

- Input will be given by *cin*

  Enter the number of interations for the loop: N

  Enter the coefficients of the equation [y = ax^3 + bx^2 + cx + d]

  a_0

  b_0

  c_0

  d_0

  x N times

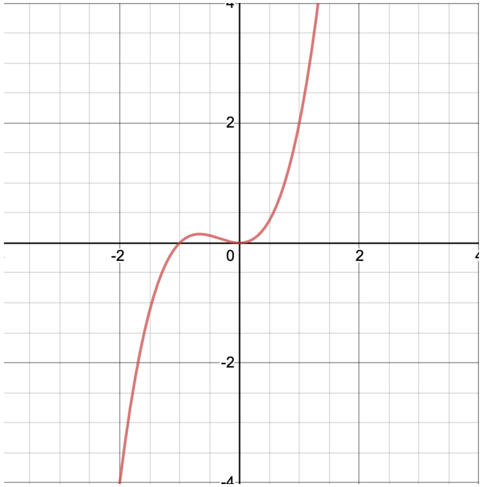
- If the answer comes out to be an imaginary unit,

  print “Not Available.”

- If is a single answer, print the only number.
- If the answer is two numbers, print to numbers, *starting from the bigger one followed by the smaller one*.
  - One spacing between the numbers

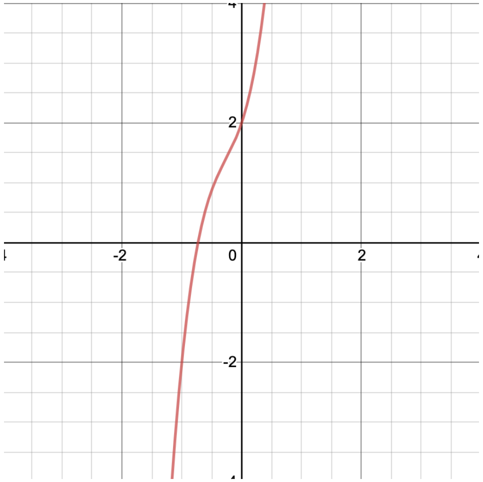
    Ex) 5.000 1.000

Problem 1



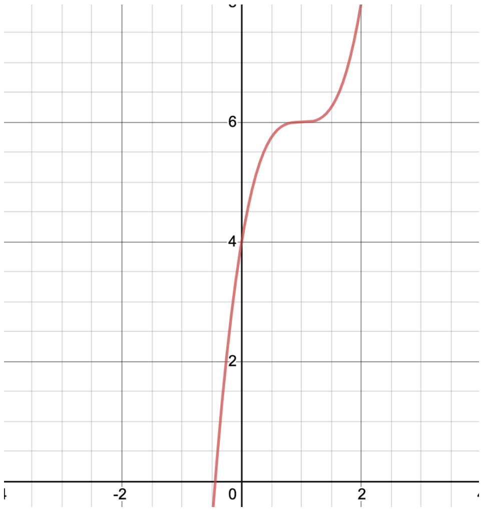
$y = x^3 + x^2$

Gradient is zero when x = 0 and -0.667



$y = 5x^3 + 4x^2 + 3x + 2$

No point is of gradient zero

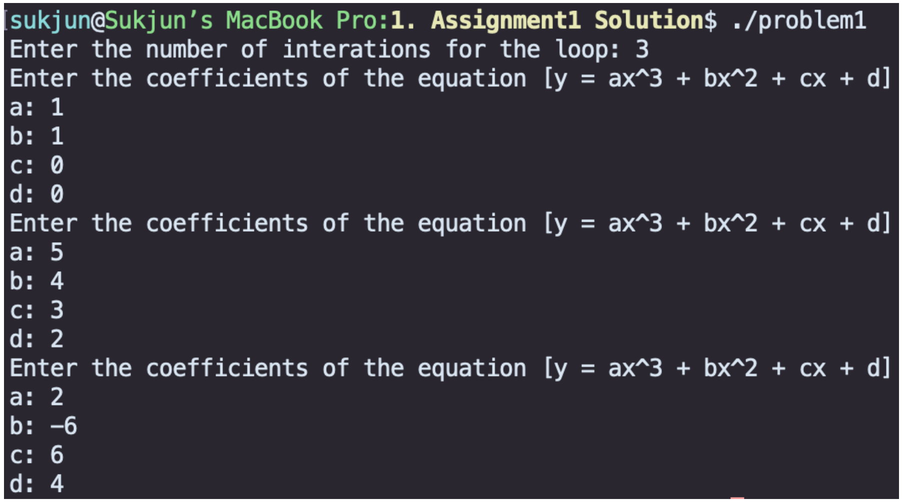


$y = 2x^3 - 6x^2 + 6x + 4$

Gradient is zero only when x = 1.0

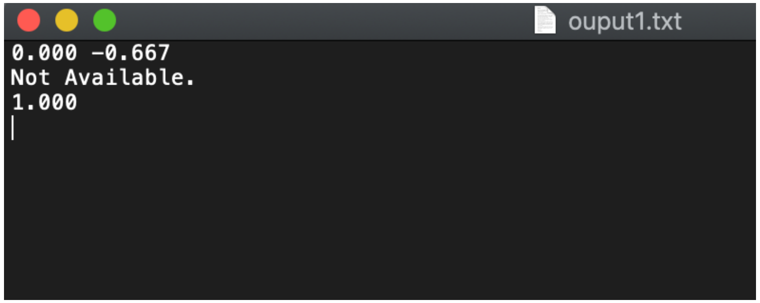
Problem 1

Input



```
sukjun@Sukjun's MacBook Pro:1. Assignment1 Solution$ ./problem1
Enter the number of interations for the loop: 3
Enter the coefficients of the equation [y = ax^3 + bx^2 + cx + d]
a: 1
b: 1
c: 0
d: 0
Enter the coefficients of the equation [y = ax^3 + bx^2 + cx + d]
a: 5
b: 4
c: 3
d: 2
Enter the coefficients of the equation [y = ax^3 + bx^2 + cx + d]
a: 2
b: -6
c: 6
d: 4
```

Output



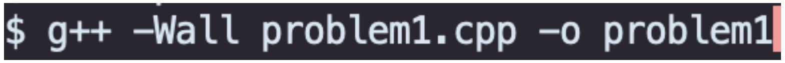
```
0.000 -0.667
Not Available.
1.000
```

Input for problem1 will not be read by file i/o.

input1.txt is given to show how the values will be given by *cin.*

output1.txt is created if ran the code

Compilation Code

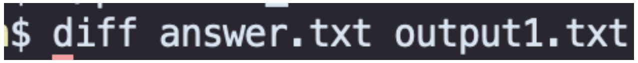


```
$ g++ -Wall problem1.cpp -o problem1
```

We will compare your output and the answer by diff command

It's good if you have nothing shown if typed the command

https://www.geeksforgeeks.org/diff-command-linux-examples/



```
$ diff answer.txt output1.txt
```

Problem 2

- Print a pyramid of stars by the rule given below
- Rule:

  The top row starts from a star at the center

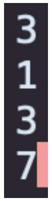
  The pyramid is of N number of lines

  The width of the pyramid increases by one per both left and right

- Input will be given by *input2.txt*
  - https://stackoverflow.com/questions/7868936/read-file-line-by-line-using-ifstream-in-c
- No need to worry. Use problem2.cpp skeleton code
- Use problem2.cpp skeleton code
- Output the results by printing in the terminal via *cout* not by file i/o

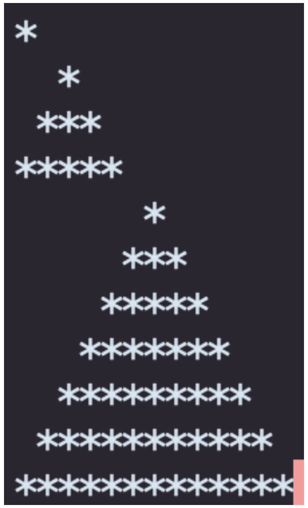
Problem 2

Input



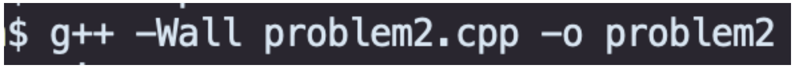
```
3
1
3
7
```

Output

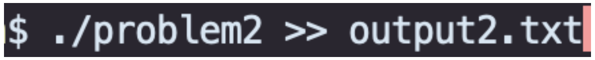


```
*
  *
 ***
*****
      *
     ***
    *****
   *******
  *********
 ***********
*************
```

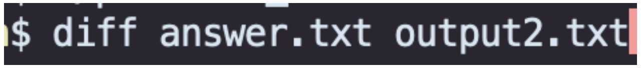
- When ran the code, the printed text *(by cout)* in the terminal can be recorded in a file by ‘>>’ command
- We will score the results by saving your programs printed texts by ‘>>’, and compare by ’diff’ command



```
$ g++ -Wall problem2.cpp -o problem2
```



```
$ ./problem2 >> output2.txt
```



```
$ diff answer.txt output2.txt
```

Problem 3

- Input : *input3.txt* Use problem3.cpp skeleton code
- You start from $S dollars and your asset increases by P% each day
- You record how much money you have each day
- You are targeting to make $T dollars and the day you achieve money more the $T dollars, the process terminates
- At last, you calculate the total income you made
- Exit the loop when money is *equal or greater* than the target
- Output the results by printing in the terminal via *cout* not by file i/o
- Print by precision of 3

Output format)

Day #1: _____

Day #2: _____

…

Final income: _____

Problem 3

- The input format is as below

N

S_0

P_0

T_0

S_1

P_1

T_1

…

S_{N-1}

P_{N-1}

T_{N-1}

N : number of loops

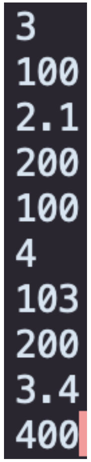
S : start money

P : profit percentage

T : target money

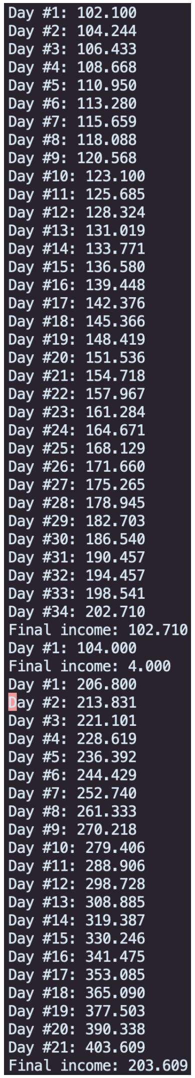
Problem 3

Input



```
3
100
2.1
200
100
4
103
200
3.4
400
```

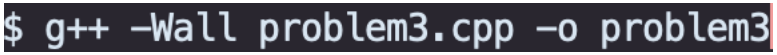
Output



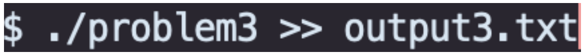
```
Day #1: 102.100
Day #2: 104.244
Day #3: 106.433
Day #4: 108.668
Day #5: 110.950
Day #6: 113.280
Day #7: 115.659
Day #8: 118.088
Day #9: 120.568
Day #10: 123.100
Day #11: 125.685
Day #12: 128.324
Day #13: 131.019
Day #14: 133.771
Day #15: 136.580
Day #16: 139.448
Day #17: 142.376
Day #18: 145.366
Day #19: 148.419
Day #20: 151.536
Day #21: 154.718
Day #22: 157.967
Day #23: 161.284
Day #24: 164.671
Day #25: 168.129
Day #26: 171.660
Day #27: 175.265
Day #28: 178.945
Day #29: 182.703
Day #30: 186.540
Day #31: 190.457
Day #32: 194.457
Day #33: 198.541
Day #34: 202.710
Final income: 102.710
Day #1: 104.000
Final income: 4.000
Day #1: 206.800
Day #2: 213.831
Day #3: 221.101
Day #4: 228.619
Day #5: 236.392
Day #6: 244.429
Day #7: 252.740
Day #8: 261.333
Day #9: 270.218
Day #10: 279.406
Day #11: 288.906
Day #12: 298.728
Day #13: 308.885
Day #14: 319.387
Day #15: 330.246
Day #16: 341.475
Day #17: 353.085
Day #18: 365.090
Day #19: 377.503
Day #20: 390.338
Day #21: 403.609
Final income: 203.609
```

Check output3.txt for finer view

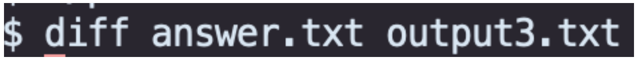
Same as problem 2



```
$ g++ -Wall problem3.cpp -o problem3
```



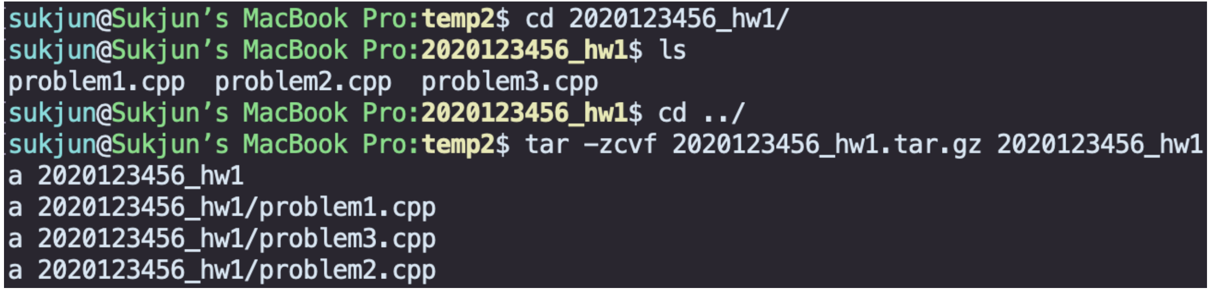
```
$ ./problem3 >> output3.txt
```



```
$ diff answer.txt output3.txt
```

Submission

- Zip the folder by following steps correctly



```
sukjun@Sukjun’s MacBook Pro:temp2$ cd 2020123456_hw1/
sukjun@Sukjun’s MacBook Pro:2020123456_hw1$ ls
problem1.cpp  problem2.cpp  problem3.cpp
sukjun@Sukjun’s MacBook Pro:2020123456_hw1$ cd ../
sukjun@Sukjun’s MacBook Pro:temp2$ tar -zcvf 2020123456_hw1.tar.gz 2020123456_hw1
a 2020123456_hw1
a 2020123456_hw1/problem1.cpp
a 2020123456_hw1/problem3.cpp
a 2020123456_hw1/problem2.cpp
```

- studentId_hw1.tar.gz
  - Ex) 2020123456_hw1.tar.gz

- There is going to be reduction of points if not following the folder hierarchy as well
- If unzipped your submission .tar.gz file should follow the folder hierarchy below

Current directory

- studentId_hw1.tar.gz
- studentId_hw1
  - problem1.cpp
  - problem2.cpp
  - problem3.cpp

Questions

- Use oop20202@gmail.com for questions

- We are not going to answer
  - Questions sent to TAs’ personal mails
  - Questions not making sense
  - Questions related to the algorithm for solving the question
  - Questions you can infer the answer if read this file thoroughly
  - Questions you can simply solve by googling
    - Ex) how do I make a folder on ubuntu?

Appendix

- File I/O

```
#include <fstream>
ofstream outfile;
outfile << "Hello, World!\n"; // writing Hello,World! into the file

outfile.close(); // should close the file before terminating the process

ifstream infile("input.txt");
infile >> number; // reading the first digit written in input.txt

infile.close(); // should close the file before terminating the process
```

https://stackoverflow.com/questions/7868936/read-file-line-by-line-using-ifstream-in-c

Appendix

- Zipping and unzipping the folder by tar command
  - https://linuxize.com/post/how-to-extract-unzip-tar-gz-file/
  - https://www.cyberciti.biz/faq/how-do-i-compress-a-whole-linux-or-unix-directory/