Object-Oriented Programming
Fall 2015

Project 1

Seon Joo Kim

Outline

- Project Overview
- Programming Problems
- Requirements & Assumptions
- Deliverables, due-date and submission

Project Overview

- Structure of the program
  - [Main Menu]
    - [Problem 1]
      - Choose shape to draw
      - Draw
    - [Problem 2]
      - Calculate factorial
      - Calculate combination
    - [Exit]

Project Overview

- Main menu

```
################################
Main Menu
Enter your command!
################################

1. Problem1
2. Problem2
0. Exit

Command  >>
```

Project Overview

- Main menu example

```
[csi2102@localhost Project1]$ ./out
##############################
Main Menu
Enter your command!
##############################

1. Program1
2. Program2
0. Exit

Command >>
```

Project Overview

- Main menu (termination)

```
##################################
Main Menu
Enter your command!
##################################

1. Problem1
2. Problem2
0. Exit

Command >> 0

Program is terminating
```

User input is underlined.

Project Overview

- Main menu example (termination)

```
##############################
Main Menu
Enter your command!
##############################

1. Program1
2. Program2
0. Exit

Command >> 0

Program is terminating

[csi2102@localhost Project1]$
```

Project Overview

- Main menu (wrong input)

```
##################################
Main Menu
Enter your command!
##################################

1. Problem1
2. Problem2
0. Exit

Command >> a

Wrong Input. Please choose one of the above options.

##################################
Main Menu
…
```

Project Overview

- Main menu example (wrong input)

```
##############################
Main Menu
Enter your command!
##############################

1. Program1
2. Program2
0. Exit

Command >> a

Wrong input. Please choose one of the above options.

##############################
Main Menu
Enter your command!
##############################

1. Program1
2. Program2
0. Exit

Command >>
```

Tip

- How to write menu (for reference)

```
while(1)
{
     cin>>option;

     switch(option)
     {
          case ‘1’:
               // code for problem 1
               break;
          case ‘2’:
               // code for problem 2
               break;
          case ‘0’:
               // message about termination
               return 0;
          default :
               // Error message
               continue; // can be skipped
     }
}
```

Use variable type ‘char’

Outline

- Project Overview ✓
- Programming Problems
- Requirements & Assumptions
- Deliverables, due-date and submission

Programming Problems 1

- Write a program that draw some shapes with ‘*’ character within 9*5 area.

- To draw shapes, you must use flow of control. Printing ‘*’ character manually is not allowed.

- User can choose type of shape to draw among following shapes:

  Rectangle, Triangle, Inverted Triangle, Letter ‘H’

Programming Problems 1

- Example
  Rectangle, Triangle, Inverted Triangle, Letter ‘H’

```
Choose shape >> 1

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

```
Choose shape >> 2

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

```
Choose shape >> 3

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

```
Choose shape >> 4

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

Programming Problems 1

```
################################
Main Menu
Enter your command!
################################

1. Problem1
2. Problem2
0. Exit

Command >> 1

*Drawing a shape
(1-Rectangle, 2-Triangle, 3-Inverted Triangle, 4-Letter ‘H’, 0-Back to Main Menu)
Choose shape >> 3

*********
 *******
  *****
   ***
    *

Choose shape >> 0

Back to main menu

################################
Main Menu …
```

Programming Problems 1(Example)

```
Command >> 1

*Drawing a shape
(1-Rectangle, 2-Triangle, 3-Interveted triangle, 4-Letter ‘H’, 0-Back to main menu)
Choose shape >> 1

*********
*********
*********
*********
*********

Choose shape >> 0

Back to main menu

###############################
Main Menu
Enter your command!
###############################

1. Program1
2. Program2
0. Exit

Command >>
```

Programming Problems 2

- Write a program that is called the Simple calculator.
- During calculation, you must use flow of control.
- User can choose type of mathematics functions :
  -> Factorial or Combination
- User can input variables of functions.
  -> n or k
- Example:

$$n! = \prod_{k=1}^{n} k$$

$$n! = \begin{cases} 1 & \text{if } n = 0, \\ (n-1)! \times n & \text{if } n > 0. \end{cases}$$

0! = 1, 1! = 1, 2! = 2, 3! = 6, 4! = 24

Factorial

$$\binom{n}{k} = \frac{n!}{k!(n-k)!},$$

$$\binom{52}{5} = \frac{52 \times 51 \times 50 \times 49 \times 48}{5 \times 4 \times 3 \times 2 \times 1} = 2,598,960.$$

Combination

Programming Problems 2(Example)

```
…
Command  >> 2

*Choose Function of calculator
(1-Factorial, 2-Combination, 0-Back to main menu)
Choose function >> 1

Input of N [n!] = 5
5! is 120

Choose function >>  2

Input of N = 5
Input of K = 2
5C2 is 10

Choose function >> 0

Back to main menu

##############################
…
```

```
int nK, nR;
cin>>nK;//Assigned 5
cin>>nR;//Assigned 2

Or

int nV[2];
cin>>nV[0]; //Assigned 5
cin>>nV[1]; //Assigned 2
```

Programming Problems 2(Example)

```
Command >> 2

*Choose function of calculator
(1-Factorial, 2-Combination, 0-Back to main menu)
Choose function >> 1

Input of N [n!] = 2
2! is 2

Choose function >> 2

Input of N = 5
Input of K = 2
5C2 is 10

Choose function >> 0

Back to main menu

##############################
Main Menu
Enter your command!
##############################
```

Outline

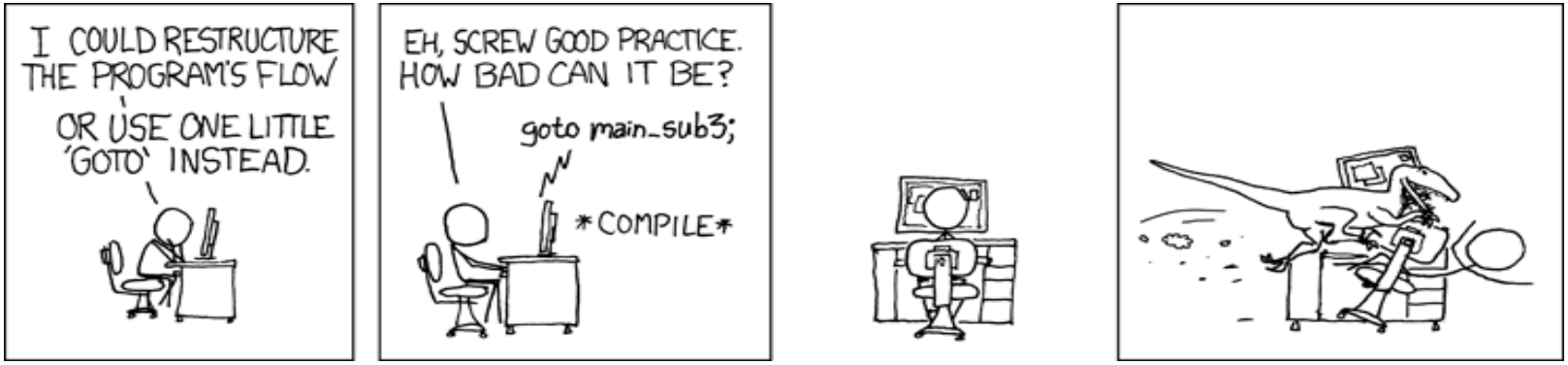
- Project Overview ✓
- Programming Problems ✓
- Requirements & Assumptions
- Deliverables, due-date and submission

Requirements & Assumptions

- You should use "Flow of Control" (for, while, switch, etc)
- User Inputs may not be valid. Exception handling required.
  - If the input cannot be understood, ask again.
- User Input is always one character for each question.
  - Except for taking inputs of 'n' and 'k' in problem2.
  - For 'n' and 'k', you can assume that user input is always valid.
- Reasonable comments in English(Important!)
- Reasonable Indentation(Important!)

Requirements & Assumptions

- Your program should work correctly in Linux Environment.
- Use ‘unsigned long long’ variable type for problem2. (In that case we can obtain a result up to 20! = 2,432,902,008,176,640,000)
- Do not use ‘goto’ statement. (It’s better not to use it unless it’s necessary.)



Outline

- Project Overview ✓
- Programming Problems ✓
- Requirements & Assumptions ✓
- Deliverables, due-date and submission

Marking Criteria and Plagiarism

- **Marking Criteria**
  - Score is only given to programs that compile and produce the correct output.
  - Points are deducted for programs that produce compiler warnings. Hint: use the –Wall command-line parameter to eliminate all warnings.
  - Points deductions on programming style: provide comments in your code and use proper indentation of lines.
  - Please pay particular attention to the requested output format of your programs. Deviating from the requested output format results in points deductions.

- **Plagiarism (Cheating)**
  - All submissions are checked for plagiarism.
  - Once detected, no score will be given for the lab to all students involved in the plagiarism incident.

Deliverables

- This time, your are required to prepare only one file for this lab.
  - For programming problems: project1.cpp

Warning: you will lose points if the file name is not proper!

Archiving the deliverables

- Please zip the all files of your submission to a single archive:

  $ tar –jcvf <student_id>.tbz2 project1.cpp

- To make sure, you can extract the archive file with following command:

  $ tar –jxvf <student_id>.tbz2

- Please note that in the above command, all must be typed in on a single line!
  - The shell will wrap-around the text for you ☺

Submitting your archive

- You are asked to upload your archive on YSCEC.
  - **<student_id>.tbz2**
- Due date: MONDAY, September 14, 23:55.
- For instructions on how to upload a file on YSCEC, please see the Lecture Note on Linux installation.

Outline

- Project Overview ✓
- Programming Problems ✓
- Requirements & Assumptions ✓
- Deliverables, due-date and submission ✓