Code Guidelines

If you do not follow these rules, points will be taken off your assignment. Do not edit your output. If you change your program, you need to compile and run your program to generate ALL the output again. Turn in all parts of your output from your final run, do not only turn in the output from any previous run. If your program does not produce any output, comment that in your program listing. Homework without output will have points deducted. Do not scribble anything on your program listing, only the code will be graded.

Tips

The name of the program must be the same as the file name of your primary .java or .cpp file.

Save your new file using .java or .cpp extension before typing your code so that the IDE will highlight the keywords and indent the program correctly for you.

Requirements

- Include a comment block at the beginning of the program using the following format.

```
/*
 * Name:
 * Section: COSC/ITSE ####
 * Homework PG ### P ##
 * Description
 */
```

- Capitalize every word in program names / class names. Examples: JavaProgram, CppProgram
- Use meaningful variable and constant names.
- Begin variable names with a lower case letter using the three-letter prefix and capitalize internal words. Example: fltTotalAmount. Letter prefix are as follows: int – integer, flt – float, cls – class, dbl – double, chr – character, str – String, bol – Boolean, and obj – object.
- Use named constants appropriately in your program. Use all uppercase for named constants be sure to include the three-letter prefix.
- Use blank lines to separate different sections of your code.
- Put a space before and after each operator.
- Indent your code correctly. Place opening braces as far left as possible without obscuring the previous opening brace. Indent all code between the opening and closing braces 4 spaces to the right of the braces. Match the closing brace with the opening brace.
- Do not type beyond your screen (max 70 characters per line); otherwise your printed code will not look correctly indented. Make sure your printed code does not go off the page. If a line is too line to fit on a line of the printed page, use multiple lines and indent subsequent lines appropriately.
- Comments are required for each class and method.
- Use local variables, parameters, and instance variables appropriately.

How To Turn in Your Programs (except GitHub)

- Run your program.
- Select the entire project folder with source files and resulting java classes or cpp application and compress them using winzip, or other zip compression program. Name the resulting compressed file with your initials, the letters ‘pg’, the page number, the letter ‘p’ and the problem number. Or if there is no page number then enter your initials and the main class name. Example file name: crmpg257p2.zip. *Note:* You should also use this naming convention for your main program unless otherwise specified by the problem directions.
- If you are have bugs in your program and your are turning it in use initials, page number, problem and add the word 'bug'. Example: crmpg257p2bug.zip. You can use 'help' if you need help. Also use 'fix' if it is a bug fix or revision.
- Email to your instructor the project as an attachment or submit in canvas or submit to github. Be sure to include the correct subject in the email.

How To Print Your Output and Program

- Run your program.
- To print your output, highlight all the output in the DOS window, click on the icon in the top left corner of the DOS window’s title bar, select Edit->Copy Enter. Create a comment block in your primary source document and paste into the comment block. Do not change or add anything.
- To print your program, you need to ensure that the program is displayed in the code pane (by clicking on the program’s name in the document selector in the left pane) and then select File->Print.
- If the program is graphical in nature make sure that you take a screen shot and include that in your project when turning it in.

Grading Scale

You will earn 10 points for each of the following items:

- Coding Guidelines followed
- Programming task completed
- Flowchart/Pseudo Code
- Comments
- Output
- Algorithm
- Best possible use of language
- Input
- Use of appropriate class material you have been exposed to
- Good programming practices