CS 135 - Computer Science I : Spring 2023
Programming Assignment 02b (**pa02b.cpp**)

Limitations

Please refrain from using advanced C++ code techniques in this assignment. Limit commands to those discussed in the current and previous chapters, labs, and lectures:

▶ **C++ Command Limit: Do not use commands beyond Chapter 02**.

Students are NOT allowed to collaborate on this programming assignment. However, you can ask a teaching assistant or tutor for help. Do not share code. If you get help or discuss this assignment with another instructor, include their name(s) in the header comments.

Unless instructed to do so, do not use the following C++ commands:

- `exit()`
- `continue`
- `break` (unless it's in a `switch` statement)
- `goto`

All functions must have only ONE `return` statement, which must be located at the end of the function. This rule includes `main()` .
(see https:/dbrodersen.net/codeRubric_cs135 for a submission checklist.)

Your code must adhere to the course style guide to get credit for this programming assignment.
(see https:/dbrodersen.net/comments for code formatting standards and how to comment your code.)

▶ Your output must match the provided sample interactions, if any.
▶ Never use TAB characters in source code or program output.

Please submit questions regarding this assignment at least 24 to 48 hours before it is due to allow time for a response that you can implement.

Programming Assignment: Students may NOT collaborate with each other

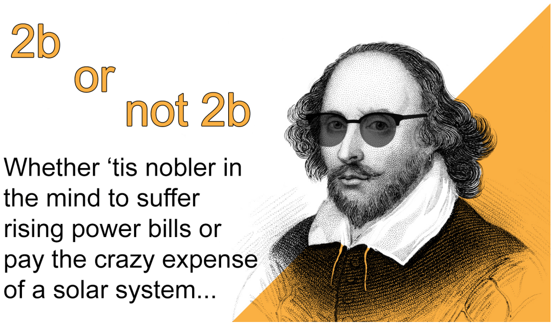
To complete this assignment you must first...

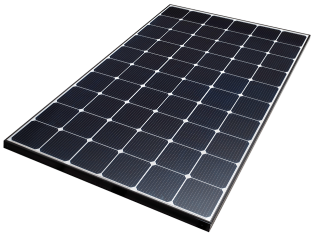
▶ Read the text through Chapter 02 , and completed the lab assignment.
▶ Complete the previous chapter labs and programming assignments
▶ Study the sample solutions for the previous assignments

The due date for this assignment is posted in Canvas.

Section: 1 Problem Description

A solar power system is used to generate electricity from the sun. The amount of energy generated depends on the amount of sunlight received, the size of the solar panel, and the efficiency of the system.
You have been hired by a solar energy company to write a program that calculates the amount of electricity generated by a solar panel system. The program should take into account the size of the panel in square meters, the efficiency of the system, and the amount of sunlight received in hours. The program should then calculate the amount of electricity generated in kilowatt-hours (kWh).


2b
or
not 2b
Whether 'tis nobler in the mind to suffer rising power bills or pay the crazy expense of a solar system...



▶ The formula for calculating the amount of electricity generated in watt-hours is:

$$E = A * e * sl * c * 0.001$$

Where:

- E is the electricity generated (Wh)
- e is the specified efficiency of the panel.
- A is the area of the panel in m^2
- sl is the amount of sunlight exposure in hours
- c is the panel generation capability (m^2 per hour) = 300
- *0.001* is a conversion factor to kilowatt-hours (kWh)

Programming Assignment: Students may NOT collaborate with each other

Section: 2 Program Specifications

- The program should prompt the user for the dimensions of a panel, efficiency, and amount of sunlight.
- The input for the dimensions, efficiency, and sunlight should be of type double.
- The program should use the formula above to calculate the amount of energy generated.
- The result should be output to the user in kilowatt-hours (kWh)
- The program should use a `static_cast` to convert the kilowatt-hours value to an integer before printing the result, to round the result down to the nearest whole number.
- The program should use a string variable to store a message that explains the result.
- The program should include appropriate comments and meaningful variable names that are documented.

Section: 3 Assignment

1. Start by creating a new C++ project and write a program that prompts the user for the size of the panel, efficiency, and amount of sunlight. To create a C++ project, begin my creating a directory for your source code file followed by completing all your work in that directory.
2. Use the input values to calculate the amount of electricity generated using the formula above.
3. Use a `static_cast` to convert the results of the calculation to an integer and round down to the nearest whole number.
4. Use a string variable to store a message that explains the result, e.g, "kilowatt-hours (kWh) of energy per panel per day". (See sample interaction.)
5. Output the message to the user along with the expected number of kilowatt-hours.
6. Test the program to ensure it works as expected.
7. Document the program with appropriate comments and meaningful variable names that each have their own documentation describing the purpose of the variable.

Programming Assignment: Students may NOT collaborate with each other

Section: 4 Sample Interaction

Assuming you compiled using the output option to name the output file to pa02b, here are two sample interactions for your program once it is compiled and running correctly.

Section: 4.1 Example 1

Text in red is command invocation. Highlighted blue text represents user input:
note:

- *the $ is not part of the command. It represents the bash prompt.*
- *the output file name has no extension.*

Sample Interaction

```
$ ./pa02b
Enter the length and width of a panel in meters, separated by a space: 1.0 1.0
Enter the efficiency of the system as a decimal (0 to 1): 1.0
Enter the amount of sunlight received in hours: 10.0

kilowatt-hours (kWh) of energy per panel per day: 3
$
```

Section: 4.2 Example 2

Sample Interaction

```
$ ./pa02b
Enter the length and width of a panel in meters, separated by a space: 1.6 1.0
Enter the efficiency of the system as a decimal (0 to 1): 0.90
Enter the amount of sunlight received in hours: 5.2

kilowatt-hours (kWh) of energy per panel per day: 2
$
```

Programming Assignment: Students may NOT collaborate with each other

Helpful commands for this assignment

Bellagio Linux Console (Terminal)

```
$ mkdir -p ~/cs135/pa02b/
$ cd ~/cs135/pa02b/
$ nano pa02b.cpp
$ geany pa02b.cpp
$ cpplint pa02b.cpp
$ pclint pa02b.cpp
$ cppcheck pa02b.cpp
$ aspell -c pa02b.cpp
$ astyle pa02b.cpp
$ g++ $CXXFLAGS pa02b.cpp -o pa02b
$ ./pa02b
$ man linux command
$ cppman CPP command
```

Where:

`mkdir -p`
creates a directory and its path

`nano`
ASCII text editor available on Bellagio for editing program files

`geany`
IDE for working on programs (works best with x2go)

`cpplint`
Open Source Linter (static code analyzer) by Google

`pclint`
Commercial Linter by Gimp Suit Software

`cppcheck --language=c++ --std=c++14`
Performs a static check for bugs and undefined behavior

`aspell -c`
spell checker

`astyle`
artistic style code indenter / formatter / beautifier

`g++ $CXXFLAGS`
C++ compiler. The option **-o** will allow you to name the executable file

`man` Immediate online help on a Linux command

`cppman`
Immediate online help on a C++ command

Programming Assignment: Students may NOT collaborate with each other

Turn in pa02b

To receive credit for your work:

- Files must be correctly named
- Files must be correctly submitted before the due date and time
- The program must compile on the bellagio server.
- The program must not crash when launched or while running
- A correctly formatted header comment must be included
  (see: `https://dbrodersen.net/comments`)
- Code must be properly commented
  (see: `https://dbrodersen.net/comments`)
- Code must be properly formatted, such as proper indentation, etc
  (see video examples and sample solutions)
- TABs must not be used in the source code or output of the program

Note: *It's really important that you implement your assignments exactly as written. No extra bells and whistles, please. I've written a shell script that:*

- ▶ *uses cpplint and pclint along with custom commands that go through the mechanics of your code to make sure syntax, variable names, functions, procedures, and spelling are all correct.*
- ▶ *runs the assignment using the sample data exemplified in the assignment description and compares your program screen and file output with the expected output*
- ▶ *runs the assignment a second time using data other than that provided in the sample solution*

In this way, I am able to determine if the logic and calculations in your program are correct. Lastly, I look at the code myself, scanning for code styling, proper commenting, and efficiency.

When you include extra features, the script flags them and deducts points.

Programming Assignment: Students may NOT collaborate with each other

Labs and programming assignments are NOT turned in using Canvas. Use the command below while logged into *Bellagio* to turn in your assignment. Cut and paste will not work with this command. You must type it in yourself. Note: you must be in the directory where pa02b.cpp exists for this command to work.

Bellagio Linux Console (Terminal)

```
$ turnin -c cs135-dbrodersen -p pa02b -v pa02b.cpp

— results of a successful submission —
Submitted files:
pa02b-username
pa02b-username/pa02b.cpp
$
```

The Bellagio **turnin** command submits assignments to be graded. The drop box for the assignment closes promptly at the due date/time. If the project is enabled and you want to resubmit your assignment, you may do so by re-running the `turnin` command.

Note: New submissions overwrite any previous submissions.

Syntax: `turnin -c` *course-name* `-p` *project-id* `-v` *file name(s)* separated by spaces.

Option	Purpose
`-c`	**Course**. Sets the submission course to which we'll submit our assignments. *Required*.
`-p`	**Project**. Sets the project to which we'll submit our assignments. The project id is the name of the file without the extension.
`-v`	**Verbose**. Prints a list of submitted files once they have been submitted.
`-h`	**Help**. Print a help message.
`-l`	**List**. Prints a list of projects, along with whether or not they are enabled and shows which project is the default project.

Programming Assignment: Students may NOT collaborate with each other