Project [7]: Pointers

Due Tuesday, 11/6/2018, 11:59 pm

Project Goals

The goal of this project is to:

1. Get students familiar with the use of pointers

Important Notes:

1. **Formatting:** Make sure that you follow the precise recommendations for the output content and formatting: for example, do not change the text of the problem from "`Please type the real component: `" to "`Enter real part: `". Your assignment will be auto-graded and any change in formatting will result in a loss in the grade.
2. **Comments:** Header comments are required on all files, for each function, and recommended throughout the rest of the program. Points will be deducted if no header/function comments are included.
3. **Restriction:** The use of `goto` statements anywhere within this program is prohibited. Points will be deducted if `goto` is used.

Problem 1

Write a program that adds, subtracts, and multiplies complex numbers. This program will prompt the user for the choice of operation, read in the complex numbers, perform the operation, and print the result. Your program must use the six functions described below.

At the beginning, the program should output the following:

```
Complex Number Arithmetic Program:
```

At every iteration, your program should ask the user for an option as follows:

```
1) Add two complex numbers
2) Subtract two complex numbers
3) Multiply two complex numbers
4) Quit
Choose an option (1 - 4):
```

Each option should function as follows:

- Option 1: Read in two imaginary numbers, add them together, and print out the result.
- Option 2: Read in two imaginary numbers, subtract the second one from the first, and print the result.
- Option 3: Read in two imaginary numbers, multiply them together, and print the result.
- Option 4: Print the message below and end the program
    - "`Bye!`"

Your program must use and implement the following functions:

- `add`: This function returns nothing and takes as parameters four floats (The real and imaginary parts of two imaginary numbers). The function also takes as parameters two float pointers (pointers to the imaginary and real portion of the result). This function adds the two imaginary numbers. This real part of the result is pointed to by the first pointer and the imaginary part of the result if pointed to by the second pointer. The function prototype is as follows:

    - `void add(float real_part_1, float imaginary_part_1, float real_part_2, float imaginary_part_2, float* real_result, float* imaginary_result)`
- `subtract`: This function returns nothing and takes as parameters four floats (The real and imaginary parts of two imaginary numbers). The function also takes as parameters two float pointers (pointers to the imaginary and real portion of the result). This function subtracts the second imaginary number from the first. This real part of the result is pointed to by the first pointer and the imaginary part of the result is pointed to by the second pointer. The function prototype is as follows:
    - `void subtract(float real_part_1, float imaginary_part_1, float real_part_2, float imaginary_part_2, float* real_result, float* imaginary_result)`
- `multiply`: This function returns nothing and takes as parameters four floats (The real and imaginary parts of two imaginary numbers). The function also takes as parameters two float pointers (pointers to the imaginary and real portion of the result). This function multiplies the two imaginary numbers. This real part of the result is pointed to by the first pointer and the imaginary part of the result if pointed to by the second pointer. The function prototype is as follows:
    - `void multiply(float real_part_1, float imaginary_part_1, float real_part_2, float imaginary_part_2, float* real_result, float* imaginary_result)`
- `read_num`: This function returns nothing and takes as parameters two float pointers (The first pointer to the real part and the second pointer to the imaginary part of an imaginary number). This function begins by printing:

  "`Please type in the real component: `"

  The function then reads in the real component of the imaginary number and points the first pointer passed to it to that number. Then the function prints:

  "`Please type the imaginary component: `"

  The function then reads in the imaginary component of the imaginary number and points the second pointer passed to it to that number. The function prototype is as follows:

    - `void read_num(float *real_part, float *imaginary_part)`
- `read_nums`: This function returns nothing and takes as parameters four float pointers (a pointer to the real part of the first imaginary number, a pointer to the imaginary part of the first imaginary number, a pointer to the real part of the second imaginary number, a pointer to the imaginary part of the second imaginary number). The function starts by printing:
  "`Reading the first imaginary number...`"
  Then the function calls the `read_num` function to read in the first imaginary number. Then the function prints:
  "`Reading in the second imaginary number...`"
  The function then calls the `read_num` function to read in the second imaginary number. The function prototype is as follows:
    - `void read_nums(float* real_part_1, float* imaginary_part_1, float* real_part_2, float* imaginary_part_2)`
- `print_complex`: This function returns nothing and takes as parameters two floats (one for the real part of the imaginary number and one for the imaginary part of the imaginary number). This function prints out the imaginary number by printing:
  "`The operation yields` <the real part of the number> `+` <the imaginary part of the number>`i`"
  Both the imaginary and real parts of the number should be printed with 6 spaces and three numbers after the decimal point. The function prototype is as follows:
    - `void print_complex(float real_part, float imaginary_part)`

Your program should function as follows (items underlined are to be entered by the user):

```
Complex Number Arithmetic Program:
1) Add two complex numbers
2) Subtract two complex numbers
```

```
3) Multiply two complex numbers
4) Quit
Choose an option (1 - 4): 1
Reading the first imaginary number...
Please type in the real component: 1.5
Please type in the imaginary component: 3.7
Reading the second imaginary number...
Please type in the real component: 5.4
Please type in the imaginary component: 2.3
The operation yields  6.900 +  6.000i
1) Add two complex numbers
2) Subtract two complex numbers
3) Multiply two complex numbers
4) Quit
Choose an option (1 - 4): 2
Reading the first imaginary number...
Please type in the real component: 6.
Please type in the imaginary component: 1.24
Reading the second imaginary number...
Please type in the real component: 8.23
Please type in the imaginary component: 6.754
The operation yields -1.930 + -5.514i
1) Add two complex numbers
2) Subtract two complex numbers
3) Multiply two complex numbers
4) Quit
Choose an option (1 - 4): 3
Reading the first imaginary number...
Please type in the real component: 2.65
Please type in the imaginary component: 3.7
Reading the second imaginary number...
Please type in the real component: 5.4
Please type in the imaginary component: 2.31
The operation yields  5.763 + 26.102i
1) Add two complex numbers
2) Subtract two complex numbers
3) Multiply two complex numbers
4) Quit
Choose an option (1 - 4): 5
Please input a valid menu option
1) Add two complex numbers
2) Subtract two complex numbers
3) Multiply two complex numbers
4) Quit
Choose an option (1 - 4): 4
Bye!
```

Notes:

- If the user enters an invalid menu option, the program should print the following error message:

  "`Please input a valid menu option`"

- For each operation (add, subtract, multiply) you can assume the user will give valid input

Save your program as `complex_calculator.c`

Challenge for problem 1 (10 extra credit points):

Add a division operation to your calculator. The menu is as follows:

```
1) Add two complex numbers
2) Subtract two complex numbers
3) Multiply two complex numbers
4) Divide two complex numbers
5) Quit
Choose an option (1 - 5):
```

Each option must function as follows:

- Option 1-3: The same as before
- Option 4: Read in two imaginary numbers, divide the first by the second, and print the result.
- Option 5: Print the message below and end the program
    - "Bye!"

Your program can use the previous functions and add the following function:

- `divide`: This function returns nothing and takes as parameters four floats (The real and imaginary parts of two imaginary numbers). The function also takes as parameters two float pointers (pointers to the imaginary and real portion of the result). This function divides the first imaginary number by the second. This real part of the result is pointed to by the first pointer and the imaginary part of the result is pointed to by the second pointer. The function prototype is as follows:
    - `void divide(float real_part_1, float imaginary_part_1, float real_part_2, float imaginary_part_2, float* real_result, float* imaginary_result)`

Your program should function as follows (items underlined are to be entered by the user):

```
Complex Number Arithmetic Program:
1) Add two complex numbers
2) Subtract two complex numbers
3) Multiply two complex numbers
4) Divide two complex numbers
5) Quit
Choose an option (1 - 5): 4
Reading the first imaginary number...
Please type in the real component: 2
Please type in the imaginary component: -1
Reading the second imaginary number...
Please type in the real component: -3
Please type in the imaginary component: 6
The operation yields -0.267 + -0.200i
1) Add two complex numbers
2) Subtract two complex numbers
3) Multiply two complex numbers
4) Divide two complex numbers
5) Quit
Choose an option (1 - 5): 6
Please input a valid menu option
1) Add two complex numbers
2) Subtract two complex numbers
3) Multiply two complex numbers
4) Divide two complex numbers
5) Quit
Choose an option (1 - 5): 5
Bye!
```

Save your challenge separately as `complex_calculator_c.c`

Grading Rubric

Grading will be done for each problem as follows:

Correctly-named file	5%
Header comment	2%
Program compiles	5%
Correctly-reading data from terminal	28%
Correct result printed	60%

Submission details

To submit your project, you will have to use the submission script. You do this by either:

1. Working on an ECC machine
2. Working on the provided VMware
3. Secure Copying your files (See Mac Support for information)

To Submit your project:

- Have a directory called "project7"
- Save your *.c files in that directory
- To submit: **(don't type the '>' symbols)**

  > cd project7
  > submit

The submission script copies all files in the current directory to our directory. You may submit as many times as you like before the deadline, we only keep the last submission.

Academic Honesty

Academic dishonesty is against university as well as the system community standards. Academic dishonesty includes, but is not limited to, the following:

Plagiarism: defined as submitting the language, ideas, thoughts or work of another as one's own; or assisting in the act of plagiarism by allowing one's work to be used in this fashion.

Cheating: defined as (1) obtaining or providing unauthorized information during an examination through verbal, visual or unauthorized use of books, notes, text and other materials; (2) obtaining or providing information concerning all or part of an examination prior to that examination; (3) taking an examination for another student, or

arranging for another person to take an exam in one's place; (4) altering or changing test answers after submittal for grading, grades after grades have been awarded, or other academic records once these are official.

Cheating, plagiarism or otherwise obtaining grades under false pretenses" constitute academic dishonesty according to the code of this university. Academic dishonesty will not be tolerated and penalties can include cancelling a student's enrolment without a grade, giving an F for the course, or for the assignment. For more details, see the University of Nevada, Reno General Catalog.