CS 1313 010 Spring 2025 Homework #5

Quiz to be held in lecture 9:00-9:15am Mon Feb 17 2025

Please feel free to discuss these questions with your classmates, but **NOT** to copy each other.
NOTE: Except where and as explicitly permitted in writing (for example, in a Homework),
you are **ABSOLUTELY FORBIDDEN** to **COPY EVEN A SINGLE CHARACTER** from,
or to have **ANY** shared code with, **ANY** other entity,
whether a human being (regardless of whether in CS1313 or not),
a text resource, a computing resource or anything else,
whether in person, on a local computer, online or anywhere else.
It's **INCREDIBLY EASY** for us to detect such copying, so **DON'T EVEN THINK ABOUT IT!**

1. **HOW CAN YOU TELL** that a declaration statement declares a named constant?

2. **HOW CAN YOU TELL** that a declaration statement declares a variable?

3. **WHAT IS THE DIFFERENCE** between a constant and a variable? **NOTE: This question is NOT about how can you tell what a declaration statement declares.**

4. **WHAT IS THE DIFFERENCE** between a named constant and a literal constant? **NOTE: This question is NOT about how can you tell what a declaration statement declares.**

5. **WHY** are numeric literal constants in the body of a program **BAD BAD BAD?**

6. **WHY** are named constants in the body of a program **GOOD?**

7. For each of the following, **WRITE A DECLARATION STATEMENT** for a named constant representing this quantity. For each, you should choose an appropriate data type and initialization value. The name should comply with the "favorite professor" rule, and should also be a valid C identifier. Assume that `int` variables and `float` variables take 4 bytes (32 bits) each.
   (a) the seating capacity of Meacham Auditorium (in the Union)[1]

   (b) speed of light in a vacuum in meters per second[2]

   (c) height of Mount Everest in meters[3]

   (d) length of a leap year in days

8. **YES OR NO:** Are literal constants declared?

9. **WRITE** the shortest possible **VALID** C program. (Here, *valid* means acceptable to the compiler. The program does not have to be useful, nor does it have to follow any of this course's rules for programming projects.) What does it do when you run it?

[1] https://www.ou.edu/union/host-your-event
[2] https://en.wikipedia.org/wiki/Speed_of_light
[3] https://en.wikipedia.org/wiki/Mount_Everest

10. **WHAT IS THE OUTPUT** of each of these programs, for each of the following inputs? (You do not need to show the output of the greeting nor the prompt message.) Examine the programs **CAREFULLY.** If you are not confident of your answer, type in, compile and run the programs.

(a)
```
#include <stdio.h>

int main ()
{ /* main */
    const float standard_deduction = 4150.0;
    const float single_exemption   = 2650.0;
    const float tax_rate           = 0.15;
    const int   tax_year           = 1997;

    float income, tax;

    printf("I'm going to calculate the federal income tax\n");
    printf("  on your %d income.\n", tax_year);
    printf("What was your %d income in dollars?\n",
        tax_year);
    scanf("%f", &income);
    tax = (income -
           (standard_deduction + single_exemption)) *
          tax_rate;
    printf("The %d federal income tax on $%2.2f\n",
        tax_year, income);
    printf("  was $%2.2f.\n", tax);
} /* main */
```

i. **25000**

ii. **35000**

iii. **50000**

(b)
```
#include <stdio.h>

int main ()
{ /* main */
    const float standard_deduction = 4300.0;
    const float single_exemption   = 2750.0;
    const float tax_rate           = 0.15;
    const int   tax_year           = 1999;

    float income, tax;

    printf("I'm going to calculate the federal income tax\n");
    printf("  on your %d income.\n", tax_year);
    printf("What was your %d income in dollars?\n",
        tax_year);
    scanf("%f", &income);
    tax = (income -
           (standard_deduction + single_exemption)) *
          tax_rate;
    printf("The %d federal income tax on $%2.2f\n",
        tax_year, income);
    printf("  was $%2.2f.\n", tax);
} /* main */
```

i. **25000**

ii. **35000**

iii. **50000**

11. **WHY** can C only **approximate** most (mathematical) real numbers?

12. On a Linux PC under the GNU `gcc` compiler (the compiler being used in this course), **HOW MANY BITS** are in an `int`? Therefore, **HOW MANY DIFFERENT POSSIBLE VALUES** could an `int` variable represent?

13. On a Linux PC under the GNU `gcc` compiler (the compiler being used in this course), **HOW MANY BITS** are in a `float` by default? Therefore, **HOW MANY DIFFERENT POSSIBLE VALUES** could a `float` variable of the default number of bits exactly represent?

14. Consider each of these values. **MATHEMATICALLY,** does it represent an integer? **EXPLAIN.**
    (a) $344513.00000000000000000000000001$

    (b) $344513.00000000000000000000000000$

    (c) -5281023984

    (d) $1 \cdot 10^{18}$

    (e) $-6/3$

    (f) $+9/5$

15. Consider each of these numeric literal constants. **COMPUTATIONALLY,** does it represent an integer? **EXPLAIN.**
    (a) `344513.00000000000000000000000001`

    (b) `344513.00000000000000000000000000`

    (c) `-5281023984`

    (d) `1E+18`

16. **NAME THREE REASONS** why computers use both integers and real numbers.

    (a)

    (b)

    (c)

17. **WHAT IS THE DATA TYPE** of each of the following literal constants? If the item **ISN'T** a valid literal constant, mark it **INVALID** and **EXPLAIN.**

    (a) `2004982098`

    (b) `2004982098.0`

    (c) `2,004,982,098`

    (d) `2004982098-`

    (e) `--2004982098`

    (f) `-2004982098`

    (g) `-3529.3098e+10`

    (h) `-3529.3098e-10`

    (i) `0.0`

    (j) `0`

    (k) `2e-05`

    (l) `2.0e-05`

    (m) `What do you want?"`

    (n) `"Huh?`

    (o) `"Howdy do!"`

18. Numeric literal constants can be used in several ways, some of which are good programming practice and some of which are bad programming practice. **MARK** each of the following uses as either **GOOD** or **BAD**.

    (a) `const int feet_per_mile = 5280;`
    (b) `float height_in_m = 1.6;`
    (c) `rainfall_in_inches = 5;`
    (d) `degrees_fahrenheit = degrees_celsius * (9.0 / 5.0) + 32.0;`

19. For each of the following, **WRITE A DECLARATION STATEMENT** for a variable representing this quantity. For each, you should choose an appropriate data type. The name should comply with the "favorite professor" rule, and should also be a valid C identifier. **You DON'T need to initialize the variables.** Assume that `int` variables and `float` variables take 4 bytes (32 bits) each.

    (a) the number of students in CS1313

    (b) your height in lightyears (a lightyear is the distance that light travels in a year, which is about 6 trillion miles).

    (c) a spaceship's speed in inches per century, approximated to three significant figures (assume that the spaceship travels at 99% of the speed of light)

    (d) the number of books on a bookshelf

20. **WHY** are integers sometimes called *fixed point* numbers?

21. **WHY** are real numbers sometimes called *floating point* numbers?

If you use **ANY** resources other than Dr. Neeman, the TAs/graders (Basiri, Bilal), the course textbook or the materials posted on the course webpage, you **MUST** reference them on the quiz. **THIS INCLUDES CLASSMATES, FRIENDS, PROFESSORS, ONLINE RESOURCES, ETC.**