CSE 5A
Program Grading

Program Grading

Points will be assigned to programs on the following basis.
If your program crashes, uses global variables, or does NOT compile, then NO points are awarded for output, part c).

a)	Noncompliance of Program Assignment instructions:	-10	points
b)	Crashes, global variables, late, NOT found, or not compile:	0	points
c)	Runs and produces correct output for any input:	+12	points
d)	Explanatory C comments showing how it works, user friendly output:	+2	point
e)	Coding: indentation, identified names reflect usage, lower case variable names, upper case constant types, easy to read, no repetitive code	+6	points

WARNING – ZERO points for files named INCORRECTLY

Grading Input

Programs edited after the deadline will not be graded. After the due date, the input used to grade your program will be made available in the "cs5e public" / "Input" folder and appropriately named to associate with the HW1# e.g. "hw1_input.txt".

If you think an error has been made in the grading of your program, first check your program with the input file and then see the Instructor **within 7 days** after the assignment is returned. After that time, the grade stands.

Grades will be posted on the Internet at https://www.gradesource.com, listed by your name and accessed by your secret number that was emailed to you.

Warning: a regrade may result in a lower grade with detailed analysis and strict adherence to instructions.

Getting Help

a) Email is not the best way to get help. The best way to get help is to visit the lab during posted tutor hours or to visit the Piazza site for the class at https://piazza.com (see class website for link)
b) Include your CSE 5A login account name in all email sent to the tutors or instructor. Anonymous emails will be discarded.
c) Use email **ONLY** for short clarification questions. Do not email your programs and ask what is wrong with the program. If you cannot debug or figure out your program, make a printout and see either a tutor or the instructor.
d) Do not expect an immediate answer from email. We will try to answer all reasonable email questions within 48 hours. Do not send a message a day before the deadline and expect an answer in time to meet the deadline.
e) Come to discussion sections.

Style Rules for Programs

Your program must satisfy all these **coding style** rules for full credit.

1) Indent according to the examples in the handouts and book. You need not count spaces, but indent in approximately the same places.
2) Use **meaningful** names for identifiers.
3) Avoid **hard coding**   e.g.   printf("Grade %d", 100);
   and use symbolic constants   e.g.   #define MAX_SCORE 100
4) Use functions for logically distinct tasks (after they have been introduced).
5) Use local variables and pass arguments. Constant types are recommended.
6) Starting with Programming Assignment 2:
    a. Label each loop and compound statement with comments explaining what the block does. Indent the "{" the same amount as the "}", so it is easier to associate them with each other.
    b. Include a comment at the start of each function (including main) explaining what the function does, like the one at the start of the program. The header comments for functions must detail what other functions are called and what side effects, if any, the function has.
    c. Delimit these function header comments in some distinctive manner making it easy to locate visually, like the program head comment. Skipping a line before and after a logical block of statements also makes the program easier to read and keeps the grader happy.
    d. The program's header comment block includes the login name. The header block must describe the input and output of the program, as well as a general description of the program. Normally those are all the comments you need, but if something is very confusing, include any extra comments. Too many comments can make a program hard to read as one with too few. You don't want to overdo it, but if in doubt, it is safer to put it in than leave it out.
7) Your program must be clear and easy to read.

The following is an example of a program following the style rules.

```c
/*
 * Name: Jane-Joe Student
 * Login: cs5eXX
 * Date: August 12, 2018
 * File: hw0.c
 * Sources of Help:

 * This program converts from inches to feet. The user will type in one whole number.
 * Type integer is used for the variable to hold the inches value. Type double is used
 * to calculate the number of feet. The program will print out the input number of
 * inches and display the computed value in feet.
 */
#include <stdio.h>

#define INCH_PER_FOOT   12.0                 //Constant conversion factor

double inchToFeet(double inch);        //Function declaration

/*
 * Driver for function inchToFeet(). Reads user input.
 */
int main(void)
{
    int    inches = 0;                        //Initialize to zero values
    double feet   = 0;                        //Converted value

    do
    {
        printf("Enter a positive whole number in inches for conversion to feet: ");
        scanf("%d", &inches);
    } while (inches < 0);

    feet = inchToFeet(inches);
    printf(" %d inches are equal to %lf feet.\n", inches, feet);

    return 0;
}

/*
 * Function inchToFeet() computes the value of feet using a symbolic constant.
 * The function returns the number of feet as a type double.
 */
double inchToFeet(int inch)
{
    double ft;                                //Local variable, converted value
    ft = inch / INCH_PER_FOOT;
    return ft;
}
```