ECS 30 **Homework #2 (25 points)** **Fall 2013**

Due: Wednesday, October 16th. Written: 4pm in 2131 Kemper. Programs: 11:55pm using handin to cs30, p2 directory. Filenames: track.c, house.c, can.c, pop.c

Written (5 points): p. 167: 1, 2, 3, 4, 5.

1. Define top-down design and structure charts.
2. What is a function prototype?
3. When is a function executed, and where should a function prototype and function definition appear in a source program?
4. What are three advantages of using functions?
5. Is the use of functions a more efficient use of the programmer's time r the computer's time? Explain your answer.

Programming (20 points)

All programs should be able to compile with no warnings when compiled with the –Wall option, e.g. gcc –Wall track.c. You should put your name(s) in a comment on the first line of each file. Since main() should be the first function in each file, you will need to provide a prototype above main() for each function you write.

You will find my executables and cans.txt in ~ssdavis/30/p2 in the CSIF. The prompts, and output format of each program must match the examples exactly. To ensure that the actual values calculated by the programs match the examples, you should use doubles for all real variables, and ints for all who numbers. User inputs are in **`bold`**. To use functions from math.h, you must have –lm on your compile line (that is an "l" as in library) to link with the math library, e.g., gcc –Wall –lm can.c.

#1. p. 169 #6 Filename: track.c

"Four track stars have entered the mile race at the Penn Relays. Write a program that scans in the race time in minutes (**`minutes)`** and seconds (**`seconds`**) for a runner and computes and displays the speed in feet per second (**`fps`**) and in meters per second (**`mps`**). (*Hints:* There are 5280 feet in one mile, and one kilometer equals 3282 feet.) Write and call a function that displays instructions to the program user. Run the program for each star's data."

Minutes	Seconds
3	52.83
3	59.83
4	00.03
4	16.22

Additional specifications: You must use #defines for the two constants.

```
[ssdavis@lect1 private]$ track.out
This program will ask for the minutes and seconds for the time it
took for a runner to run a mile.  The program will then calculate
the feet per second and meters per second for that runner.

Minutes for the runner: 3
Seconds for the runner: 52.83
That is 22.7 feet per second, and 6.91 meters per second.
[ssdavis@lect1 private]$ track.out
This program will ask for the minutes and seconds for the time it
took for a runner to run a mile.  The program will then calculate
the feet per second and meters per second for that runner.

Minutes for the runner: 3
Seconds for the runner: 59.83
That is 22.0 feet per second, and 6.71 meters per second.
[ssdavis@lect1 private]$ track.out
This program will ask for the minutes and seconds for the time it
```

```
took for a runner to run a mile.  The program will then calculate
the feet per second and meters per second for that runner.

Minutes for the runner: 4
Seconds for the runner: 0.03
That is 22.0 feet per second, and 6.70 meters per second.
[ssdavis@lect1 private]$ track.out
This program will ask for the minutes and seconds for the time it
took for a runner to run a mile.  The program will then calculate
the feet per second and meters per second for that runner.

Minutes for the runner: 4
Seconds for the runner: 16.22
That is 20.6 feet per second, and 6.28 meters per second.
[ssdavis@lect1 private]$
```

#2. p. 169 #7   Filename: house.c

"In shopping for a new house, you must consider several factors. In this problem the initial cost of the house, the estimated annual fuel costs, and the annual tax rate are available. Write a program that will determine the total cost of a house after a five-year period and run the program for each of the following sets of data.

To calculate the house cost, add the initial cost to the fuel cost for five years, then add the taxes for five years. Taxes for one year are computed by multiplying the tax rate by the initial cost. Write and call a function that displays instructions to the program user."

Initial House Cost	Annual Fuel Cost	Tax Rate
67,000	2,300	0.025
62,000	2,500	0.025
75,000	1,850	0.020

```
[ssdavis@lect1 private]$ house.out
This program determines the total cost of owning a home for five years.
The user will enter initial cost in whole dollars, annual fuel costs in
whole dollars, and the annual tax rate as a real number.

Please enter the initial cost, fuel cost, and tax rate: 67000 2300 0.025
The total cost is $86875.00.
[ssdavis@lect1 private]$ house.out
This program determines the total cost of owning a home for five years.
The user will enter initial cost in whole dollars, annual fuel costs in
whole dollars, and the annual tax rate as a real number.

Please enter the initial cost, fuel cost, and tax rate: 62000 2500 0.025
The total cost is $82250.00.
[ssdavis@lect1 private]$ house.out
This program determines the total cost of owning a home for five years.
The user will enter initial cost in whole dollars, annual fuel costs in
whole dollars, and the annual tax rate as a real number.

Please enter the initial cost, fuel cost, and tax rate: 75000 1850 0.020
The total cost is $91750.00.
[ssdavis@lect1 private]$
```

#3. pp. 169-170 #9   Filename: can.c

"A manufacturer wishes to determine the cost of producing an open-top cylindrical container. The surface area of the container is the sum of the area of the circular base plus the area of the outside (the circumference of the base times the height of the container). Write a program to take the radius of the base, the height of the container, the cost per square centimeter of the material (**`cost`**), and the number of containers to be produced (**`quantity`**).

Calculate the cost of each container and the total cost of producing all the containers. Write and call a function that displays instructions to the user and a function that computes surface area."

Additional specifications: Instead of reading the values interactively from a user, this program should read the values for three different cans from the file cans.txt. By #including math.h, you can have access to the predefined constant for π which is M_PI.

```
[ssdavis@lect1 private]$ cat cans.txt
4.2 20.5 0.025 15
23 40 0.015 100
18.75 12.4 0.0013 1500
[ssdavis@lect1 private]$ can.out
This program calculates the cost of materials for three open-top cylindrical
containers.  It reads the radius, height, cost per square centimeter, and
quantity for each container from the file cans.txt.  Each line in cans.txt has
the format: radius height cost_per_sq_cm quantity.

Each can #1 has surface area  596.400, and the run will cost $223.65.
Each can #2 has surface area 7442.433, and the run will cost $11163.65.
Each can #3 has surface area 2565.307, and the run will cost $5002.35.
[ssdavis@lect1 private]$
```

#4. pp. 170-171 #13 Filename: pop,c

"After studying the population growth of Gotham City in the last decade of the 20th century, we have modeled Gotham's population function as

$$P(t) = 52.966 + 2.184t$$

Where *t* is years after 1990, and *P* is population in thousands. Thus, *P(0)* represents the population in 1990, which was 52.966 thousand people. Write a program that defines a function named **`population`** that predicts Gotham's population in the year provided as an input argument. Write a program that calls the function and interacts with the user as follows: "

```
Enter a year after 1990> 2015
Predicted Gotham City population for 2015 (in thousands): 107.566
```

Additional specifications: All three constants should be #defined. Nowhere else in your program should you use any numbers that do not have to do with formatting. Instead of writing the prediction to the screen, it should be written to a file named pop.txt.

```
[ssdavis@lect1 private]$ pop.out
Enter a year after 1990> 2015
[ssdavis@lect1 private]$ cat pop.txt
Predicted Gotham City population for 2015 (in thousands): 107.566
[ssdavis@lect1 private]$ pop.out
Enter a year after 1990> 2010
[ssdavis@lect1 private]$ cat pop.txt
Predicted Gotham City population for 2010 (in thousands): 96.646
[ssdavis@lect1 private]$ pop.out
Enter a year after 1990> 1990
[ssdavis@lect1 private]$ cat pop.txt
Predicted Gotham City population for 1990 (in thousands): 52.966
[ssdavis@lect1 private]$
```