PROBLEM:

(1) Write a **function** named *compute* that finds both the area and the circumference of a circle. Your function must use **pointer notation** (the address operator "&" and the indirection operator "*"), and will be a *separate* file (compute.c) from the other code (lab5.c).
(2) You are required to create a working makefile.
(3) You must edit **lab5.c** to add your name at the top, and add you name at about line 35 fprintf(data_out, "\n**Your name here.** Lab 5. \n\n");

The function prototype is:

```
/* Function to compute the area and the circumference  */
/* of a circle   */
void compute(double radius, double *area, double *cir);
```

You will need the file **lab5.c** as your main/driver program for the function. This main program will set things up, read the values from the file, and print the output sentences.
You will also need **lab5.h** and **lab5.dat**.

THE FORMULAS (in ***algebraic*** notation)(must be translated to C notation):

(1) area of a circle
$a = PI * radius^2$

(2) circumference of a circle
$c = 2 * PI * radius$

TO GET THE FILES YOU NEED:
First move to your class folder by typing: **cd csc60**
The following command will create a directory named **lab5** and put all the needed files into it below your csc60 directory.

Type: **cp -R /gaia/home/faculty/bielr/files_csc60/lab5 .**
Spaces needed: (1) After the **cp** ↑ *Don't miss the space & dot.*
(2) After the **-R**
(3) After the directory name at the end & before the dot.

After the files are in your account and you are still in **csc60**, you need to type: **chmod 755 lab5**
This will give permissions to the directory.
Next move into lab5 directory (cd lab5), and type: **chmod 644 lab5***
This will give permissions to the files.
Your new lab5 directory should now contain: lab5.c, lab5.h, lab5.dat

INPUT/OUTPUT DESCRIPTION:
The **input** is a list of varying length of type double values in the file **lab5.dat**. Each record (or line) will have one value for the radius.

The **output** is a chart showing the radius, area, and circumference of a circle.

DEFINED OUTPUT APPEARANCE:

Print statements are included in the main program, and require no changes, except your name.

```
Your name here.  Lab 5.

 Radius     Area     Circumference
--------  --------  ---------------
   3.70     43.01         23.25
   6.80    145.27         42.73
```

ONLY the first two lines are shown here. You should validate the correctness of the other lines.

ALGORITHM DEVELOPMENT - Pseudo code:

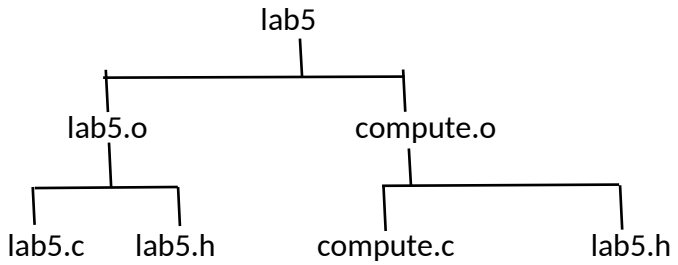
```
/*-------------------------------------------------------------------------*/
main                        /* main is given to you as lab5.c   */
   Open the data file and check for error on open.
   Open the output file and check for error on open.
   Print headers.
   while ( good data reading(fscanf) a radius )
        Call compute
        Print the radius, area, and circumference.
  Close the files.

/*-------------------------------------------------------------------------*/
/* This code, compute.c, will reside in a separate file      */
/* from the main function                                    */
/* Function to compute the area and the circumference  */
/* of a circle                                               */
/* Remember: #include "lab5.h"                               */
void compute(double radius, double *area, double *cir)

   Calculate the area
   Calculate the circumference
   Return
/*-------------------------------------------------------------------------*/
```

REMINDERS:

- Remember to put your name and Lab 5 in the comment header and in the output.
- Remember the operator for multiplication is the asterisk (*).
- Since we are already including math.h, use **M_PI** for the value of PI.
- You should examine the data file and confirm the correctness of the answer produced by your program.
- Changes: **lab5.c** to add your name; create **compute.c** in separate file; create a **makefile**
- You will also need to add **-lm** on the line that does the **gcc** in the makefile.
- The Output is going to the file **lab5.out**

CREATING A MAKE FILE: Use the slides 13-15 of 5-UNIX as a reference. Also pasted here.

The Dependency Chart of Lab5 program:



The first line should be a comment with your name.
Next: use a name like *lab5*, followed by the *.o files and the *.h file
Next: one or two tabs followed by the *.o files and the rename of the executable
You will need to add **-lm** on the line that does the **gcc** in the makefile.
Next: the *.o file name, followed by a colon,
followed by the *.c file and the *.h name
Next: compile without linking the .c file
Separate your Rules with empty lines.

PREPARE YOUR FILE FOR GRADING:
When all is well and correct,
Type: **script StudentName_lab5.txt** [Script will keep a log of your session.]
Type: **touch lab5.h** to force a recompilation
Type: **make** to compile the code
Type: **lab5** to run the program to show the output of the program
(or whatever name you used for the executable)
Type: **cat lab5.out** to see the output of your program
Type: **exit** to leave the script session

Turn in your completed session:
Go to Canvas and turn in (no zip files):

1. compute.c
2. makefile
3. lab5.c
4. your script session (StudentName_lab5.txt)

→ more on next page →

Helpful slides:

Slide 13:

/ Second pass at a makefile: */*
/ Look at its contents. We have no p2.h but it is included in light italics to show where it would be placed. */*

```
>cat makefile
# Your name
power2: power2.o compute.o  p2.h
      gcc power2.o compute.o  -o power2 -lm
power2.o: power2.c  p2.h
      gcc -c power2.c
compute.o: compute.c  p2.h
      gcc -c compute.c
```

Slide 15:

/* Helpful Comments */

When you enter **vim**, while in Command Mode, type: **:set list**

This will show the non-printable characters:

^I = tab
$ = end of line

To create a tab on athena, you may have to hit the tab key **twice** in a row.
You will know you have a correct tab when you see the ^I.

(To reverse the action of **:set list**, retype it and add an exclamation point to the end of the line. Ex: **:set list!**)
