CS 105 – Lab 3

The purpose of this lab is to practice the techniques of making choices and looping.

Log into the computer using the same procedure you used last week. Start the Terminal process. In fact, you might find it handy to have two terminals running at the same time, one where you edit code, and the other where you run programs. Hold down on the Control key, and with the mouse click on the Terminal icon. When the menu appears, select New Window.

On your USB drive or computer account, please create a new folder called lab3. All the files you create will go into this folder. Here is how you would do that: Using the "cd" command, navigate to your USB drive within the file system. When you use the "ls" command, you should see your lab2 folder from last week listed. Now, type the command "mkdir lab3" without the quotes. Then, type "cd lab3" to enter this folder.

1. As a review of what we did last week…
   a. What is the benefit of including comments in a program?

   b. Show how we would read a real number as input from the user. You need to have two statements. One to ask the user for input, and the other to put the user's value into a variable. Hint: you need to use two built-in proceudures to accomplish this.

Part 1: Fahrenheit – Celsius converter

Let's write a program that allows a user to convert between Fahrenheit and Celsius temperatures. This program will have two inputs. First, we want the program to let the user choose which way to convert, either Fahrenheit-to-Celsius or Celsius-to-Fahrenheit. The user will specify this choice by entering either the letter F or C, respectively. The second input will be the temperature. Then, the program can perform the appropriate conversion and output the result.

2. What are the types of these two inputs? Specify which one is which.

Here is the code for the program. Using nano, type it in, and save it as `convert.pas`. Please note that there are some details that you need to fill in, such as the appropriate condition to use with the if-statement, and the two conversion formulas. In your if-statement, *you will need to use quotes around the letter C or F* so that Pascal does not think that you are using this letter as a variable. Another hint: F should equal (9/5) C + 32. You can solve for C to determine the other formula.

```
(* convert.pas – Convert between Fahrenheit and Celsius
 * Purpose:  to practice the if-then-else statement.
 *)
program convert;

var
  choice, f, c : string;

begin
  writeln('Would you like to convert from F to C or from C to F?');
  writeln('Enter F if you want to convert a Fahrenheit temperature’);
  writeln('or enter C to enter a Celsius temperature.');
  write('Enter your choice (F/C):  ');
  readln(choice);

  (* Perform calculation based on whether F or C was entered. *)

  if _____________________ then
    begin
      write('Enter the Fahrenheit temperature:  ');
      readln(f);

      (* insert Celsius formula... *)
      c := _____________________;
      writeln('The equivalent Celsius temperature is ', c);
    end
  else
    begin
      write('Enter the Celsius temperature:  ‘);
      readln(c);

      (* insert Fahrenheit formula *)
      f := _____________________;
      writeln('The equivalent Fahrenheit temperature is ', f);
    end;

  writeln('Thank you for using my temperature conversion program.');
end.
```

3. Oops! There was an error in the program I gave you. How did you fix it?

4. Format the output of your program so that it prints exactly 1 decimal place, such as 68.0 degrees. Run the program both ways, to check that both formulas you entered work correctly. Ask your program for these conversions:

   14 Fahrenheit = _____ in Celsius

   25 Celsius = _____ in Fahrenheit

Part 3: Practice with multiple-choice if-statement

Let's write a simple program that works with days of the week. Create a new source file called `monday.pas`. The purpose of this program is to ask the user for a positive integer n, and then output what day of the week it is n days after a Monday. For example, if the user enters 27, the answer should be Sunday. The general format of the program is as follows. I'll describe the structure in English:

First, we ask the user to enter a positive integer, and put this answer in a variable called n.

Second, we perform the calculation n mod 7 and put the result back into n.

5. What are the smallest and largest possible values of n after we perform this calculation?

Third, we use a new variable called `day`, which will contain a string. This string will be the name of the day of the week that is n days after Monday.

Finally, we print a sentence telling the user what the value of `day` is.

To start you off, you may copy the following code to your monday.pas file. There are two things you need to add to this program. First, you need to make the appropriate assignment statement that modifies n. And second, you need to add the additional cases in the if-statement to handle all possible days of the week.

```
(* monday.pas – What day of the week is it n days after a Monday? *)
program Monday;

var
  n : int32;
  day : string;
```

```
begin
  write('Enter a positive integer:  ');
  readln(n);

  (* mod n by 7 and make this the new value of n.
   * You need to enter this assignment statement here:
   *)
  ____________________________;

  (* Use the variable called day, and set it to a string
   * such as 'Monday', 'Tuesday', etc. depending on the value of n.
   * Because there are several choices, we will make use of the
   * Pascal words if, else if, and else.
   *)

  if n = 0 then
    day := 'Monday';

  (* Insert the remaining six cases here... *)

  (* Ready for output *)
  writeln(n, ' days after Monday will be ', day);
end.
```

6. When you are ready to run your program, test it a few times with various input. Try these:

   12 days after a Monday is _____

   100 days after a Monday is _____

   365 days after a Monday is _____

   One billion days after a Monday is _____

7. How can you tell if your output (e.g. that you just wrote down above) is correct?

8. If the user entered the number 1 as input, our output will begin as follows “1 days after Monday…” This is not grammatically correct. Show how you would use an if-statement to handle this special case of n equals 1. In this case we want to print the word “day” instead of “days”.

9. When you run the program with n equals 12, the program responds by saying “5 days after Monday…” The reason why the 12 became 5 is that we performed a mod operation on n. Let’s modify the program so that it also retains the original value of n. Explain how you would do this:

   Make the necessary modification to your program, and show it to the instructor or lab aide.

Part 4: Practice with a while loop

Let’s use a while loop to help us add a series of numbers of this form:

1 + 2 + 3 + … + n

The format of this program will be simple. First, the program will ask the user for a positive integer n. Next, the program will have a while loop that counts from 1 to n. This should sound familiar because we did something similar to this in class. However, now, instead of simply counting, we want to perform a cumulative addition. So, we are going to need another variable called sum or total.

Create a new source file called sum.pas. You may use the following code to start you off. Some details have been left out, such as the initial values of count and sum, as well as the assignment statement for sum inside the while loop.

```
(* sum.pas – Find the sum of the numbers from 1 to n, where n
 * is specified by the user.
 *)
program sum;

var
  n, count, sum : int32;

begin
  write('Please enter a positive integer:  ');
  readln(n);

  count := _______;
  sum := _______;

  while count <= n do
    begin
      (* You need to insert an assignment statement to modify sum. *)
      (* And there is something else you need to do! *)

    end;

  (* output *)
  writeln('The sum from 1 to ', n, ' is ', sum);

end.
```

10. Once you are finished entering in your program, run it several times on various cases. Write down these results.

If n equals …	3	10	1414	–5
The sum is …				

11. Copy your sum.pas program into a new version called sum2.pas. Here is the command to accomplish this: "cp sum.pas sum2.pas". Change the program so that it will find the sum of the squares from 1^2 to n^2. In other words, $1^2 + 2^2 + 3^2 + \ldots + n^2$. Rephrase the wording of the final writeln statement. Compile and run your program, and test it on this input:

If n equals …	3	175	–5
The sum is …			