

INLAB 04 – CSE 1310

Note: Use **mkdir** command to create a new directory by name “lab4”. All the files for inlab 4 should be saved in this directory only. Also you need to save the source code for each checkpoint in a separate file. Names of the files should follow the format ‘seconds-chk1.c’, ‘seconds-chk2.c’ etc. After you complete a checkpoint, make a copy of that file for the next checkpoint and modify the copied file.

You will start with a simple C program. You need to connect to ‘omega.uta.edu’ server for all C programs and use ‘vi’ editor for writing the programs.

At the command prompt type the following:

vi seconds-chk1.c

/* seconds-chk1.c: This program is used to convert the value of time given in seconds to its corresponding value in minutes and seconds. It makes use of integer division and the modulus operator.

*/

#include<stdio.h>

int main(void)

{

 int input_value, /* The input to the program */
 minutes, /* Represent the time in minutes */
 seconds; /* Represent the time in seconds */

 printf(“Input the number of seconds: “);

 scanf(“%d”, &input_value);

 minutes = input_value/60; /* using the division operator */

 seconds = input_value % 60; /* using the modulus operator */

 printf(.....); /* statement to print the result */

 return (0);

}

Save the program and exit 'vi'. Work out the analysis of this program (write the pseudocode) and draw a flowchart depicting the design for the above program. Signal the grader after you have completed writing it.

Checkpoint 1

Except for the printf() statement, above program is complete. Put the appropriate code in the printf() statement to complete the program. For example, if '123' is entered as input at the prompt, the program should print the line –

123 seconds are equivalent to 2 minutes and 3 seconds.

Once the program is complete, save the program and exit the 'vi' editor. At the command prompt, compile the program using 'gcc'. If the program compiles without any errors, execute the program using 'a.out'. If the program has compilation errors, you need to fix them before executing the program. When the program prompts the user for an input, enter an integer value (e.g. 3500). The program should now display the output as three integer values.

Now type 'a.out' again and input a 'double' value (e.g. 3500.35). When you press 'Enter', you will get the result that still contains integers. Notice that the program ignores numbers after the decimal point. This is because the variables were initially declared as integer variables and this causes the resulting output value to be truncated. Signal the grader when you have finished and show the results of this checkpoint.

Checkpoint 2

Change the 'int' data type of the variables 'minutes' and 'seconds' to type 'double', but keep the data type of 'input_value' as 'int'. Also, replace '%d' with '%f' in the second printf() statement for the variables 'minutes' and 'seconds'. Compile the program and fix the errors, if there are any. Execute the program and when the program prompts for input, type an integer (e.g. 3500). Is there a change in the output? As you will see that the values of variables 'minutes' and 'seconds' are printed as 'double' data type. Signal the grader when you have finished.

Checkpoint 3

Now modify the above program so that `input_value` is converted to hours, minutes and seconds. For example, if 7384 is entered at the prompt, your program should print the line—

7384 seconds is equivalent to 2 hours, 3 minutes, and 4 seconds.

You need to modify the whole program in order to calculate the hours. At the beginning of `main()`, declare another variable `hours` of type `int` to store the value of hours. `printf()` statement should also be modified to print out the seconds in terms of hours, minutes and seconds similar to the example given above. Replace the previous calculation with the following one:

```
hours = input_value / 3600;
minutes = (input_value % 3600) / 60;
seconds = (input_value % 3600 % 60);
```

When you complete this, compile and execute the program as before and with the output on the screen, signal the grader to check your work.

Checkpoint 4

Let us modify the `'seconds-chk4.c'` source file in order to exercise arithmetic operations in C. Comment out the `printf()` statements used in the above checkpoints.

Now add the C statements to `'seconds-chk4.c'` to accomplish the following:

1. Declare three integer variables named `'num1'`, `'num2'`, and `'num3'`.
2. Prompt the user to enter values for variables `'num1'` and `'num2'`.
3. Read them individually using `'scanf()'` statements.

Perform following arithmetic operations on integer variables:

1. Multiply `'num1'` with the sum of `'num1'` and `'num2'` and assign the result to `'num2'`.
2. Subtract `'num1'` from the new value of `'num2'` and store the result in `'num2'`.
3. Divide the new value of `'num2'` by `'num1'` and assign the result to variable `'num3'`.

Use separate `printf()` statements after each operation to display the results.

Save the program and exit 'vi'. Compile and execute the program, and enter integer values to obtain the required output. Signal the grader when you get the right output.

Checkpoint 5

Make following changes to the program:

1. Change the type of variable 'num3' to 'double'. This should give a floating-point answer.
2. Make appropriate changes in the printf() statement(s) of the program to display the value of 'num3'.

Signal the grades when you get the right answer.

Checkpoint 6

Include a program header at the top of your program. The header should contain the following:

1. Your name
2. Your lab instructor's name
3. Your lab section
4. Duration of the lab
5. Lab due date
6. Brief description of object of the lab

You need to encapsulate the header within the /* and */ comment marks. Save the file and exit 'vi'. Compile the program. If it complies without error, signal the grader.

Checkpoint 7