AUBURN UNIVERSITY

CPSC 3303 Programming Assignment 1

Instructions For this assignment, you must write 3 programs in C. You cannot use built-in functions that perform these operations. In case of a doubt, check with your instructor. I chose C because it is one of the high-level languages closest to the memory/hardware. Your C programs must compile and execute on the Unix machines on the Engineering Network System. If not, no credit will be awarded to you. These machines can be accessed through the host gate.eng.auburn.edu. We will demo in class how to access these machines.

Objectives of this assignment:

- to "*dust off*" your programming skills in C
- to stress on the fact that the memory (variables) ultimately are 0s and 1s
- to practice conversion from a base (decimal, hexadecimal, or binary) to another base
- to distinguish between "numbers" and characters

What you need to do:

Programming Exercise 1 (25 points): (Call the Source code Exercise1.c)

Write a **routine/function** that takes as input a character **c** supposed to be a binary digit and returns an integer representing the value of the digit **c**. If **c** is not a binary digit, your function must return **-1**. Use this function to implement a program that prompts the user to enter a **character c** that represents a binary digit (a bit!) (Recall that **c** can be only '0' or '1'). Your program must use the **character** type for the input. If the user enters a character 'x' that is **NOT** a binary digit, you must print out the error message: "The character x is invalid: x is not a bit". If the character **c** is a bit, your main program must return an **integer** representing the **value** of the digit and print out that *value* in decimal.

Example 1: If the user enters the **character** '0', your program must print out the **value** 0.

Example 2: If the user enters the **character** '1', your program must print out the **value** 1.

Example 4: If the user enters the **character** 'B', your program must print out the error message: "The character B is invalid: B is not a bit".

Programming Exercise 2 (25 points): (Call the Source code Exercise2.c)

Write a routine/function that takes as input a character **c** supposed to be an hexadecimal digit and returns an integer representing the value of the digit **c**. If **c** is not a hexadecimal digit, your function must return **-1**. Use this function to implement a program that prompts the user to enter a **character c** that represents an hexadecimal digit (Recall that **c** can be '0', '1', '2', …., '8', '9', 'A', 'B', 'C', 'D', 'E', or 'F'). To simplify, we will use only uppercase letters for hexadecimal digits larger than 9. Your program must use the **character** type for the input. If the user enters a character 'x' that is **NOT** an hexadecimal digit, you must print out the error message: "The character x is invalid: x is not an hexadecimal digit". If the character c is an hexadecimal digit, your main program must return an **integer** representing the **value** of the digit and print out that *value* in decimal.

Example 1: If the user enters the **character** '4', your program must print out the value 4.

Example 2: If the user enters the **character** 'B', your program must print out the value 11.

Example 3: If the user enters the **character** 'E', your program must print out the value 14.

Example 4: If the user enters the **character** 't', your program must print out the error message: "The character t is invalid: t is not a hexadecimal digit".

Programming Exercise 3 (50 points): (Call the Source code Exercise3.c)
Write a routine/function that takes as input a positive integer ***n*** (0 up to 2^{32}-1) and returns a **string *s*** representing the number ***n*** in binary. Use this function to implement a program that prompts the user to enter a positive integer ***n*** (0 up to 2^{32}-1). You must write a **function** that takes as input ***n*** and returns a **string *s*** representing the number ***n*** in binary. For this assignment, you **must use the method of successive division by 2** to convert the number to binary. Ultimately, your main program must print out ***s***.
Recall that you CANNOT use built-in function to convert an integer into a string.
Example: If the user enters the number 66, your program must print out 1000010.

What you need to turn in:

- Electronic copy of your report (standalone) and source code of your programs (**standalone**). **In addition,** all programming files (source code) must be put in a zipped folder named m1-name, where *name* is your lastname. Do not use any space to name the directory (there is no space between 'm' and '1'). Recall that you must name the source programs Exercise1.c, Exercise2.c, and Exercise3.c respectively and include them in the folder. Zip the folder and post it on Canvas. In summary, your source code must be submitted twice: standalone AND in a zipped folder.
  Submit separately (not inside the zipped folder) the report as a Microsoft Word or PDF file and copies of your source code.
- Your report must:
  - State whether your code works. If it does not work, state clearly what is wrong with it.
  - Clearly explain how to compile and execute your code.
- Good writing and presentation are expected.

How this assignment will be graded:

1. The program compiles but does not produce any meaningful output (5%).
2. The program compiles and executes with major bugs (40% credit).
3. The program compiles and executes with minor bugs. The code is well-designed and commented (90% credit).
4. The program compiles and executes correctly without any apparent bugs. The code is well-designed and commented (100%).
5. If the instructor needs additional communications/actions to compile and execute your code, then a **30% penalty** will be applied.
6. **If the turn-in instructions are not correctly followed, 25% will be deducted**.