HW7

For HW7, you will write a filter program that will read the contents of a text file and will write out the ASCII values of each alphaNumeric character in a selected numeric base. First, keep prompting the user for the name of a text file until you get the name of a file that actually exists (you will have to manually create the text file using emacs before you run the program). Next prompt the user for a base, until you get a valid base (options will be base 2 to base 9). Read the contents of the file (you may want to research the `read` method, which is different from `readline`), convert each alphaNumeric character into the ASCII value in the chosen base, and write it out to a file (you will get very little credit if you write to the screen, it has to be written to a text file). If a character is not alphaNumeric (e.g. a space, a new line, $, !, @, etc.) write out `..` instead of the ASCII value. The name of the output file should be x_whateverTheNameOftheInputWas.txt (where x is the base, and whateverTheNameOftheInputWas is the name of the input file.) For example, if my input file is called myFile.txt and it needs be converted into base 4, the output file will be 4_ myFile.txt. The program should generate this name, not the user typing it in. The output file must have 5 ASCII values/`..` per line.

Question: I am tired of writing programs that have to do with base conversions. Is this the last one?

Answer: I think so.

Question: I know I can use the `ord` function to get the ASCII value in base 10, but what about the ASCII value in base 2, base 3, base 4,……, base 9?

Answer: You already converted base 10 numbers into base 2 and base 8 in HW5. Just extend the algorithm for the remaining six bases (3,4,5,6,7,9). What is very inefficient (== you will lose points) is to have ***8 different*** functions for the conversions (one for base 2, one for base 3, ….., one for base 9). Just write ***one*** function that will take in a base 10 number and the new base, and will return the number in the new base (the type of the return value (string, list, etc) is up to you).

Note: When you look at a standard ASCII table, it will only show you base 8, base 16, and base 10 (and maybe base 2) as these are the bases that are important in CS. Your program will show ASCII values in non-standard bases too.

Question: Should the program keep looping to convert multiple files?

Answer: No, after successfully converting one file, it should end.

All the logic that does the base conversion needs to be written by you. If you use any type of built-in function/method that is designed to take in a number in one base and return it in another base, you will get very little to no credit on this assignment. You can use the `ord` function to get the base 10 ASCII value of a character, but write the code to convert to the selected base.

Use the samples below as a guide for developing your program and for the format of the prompts and the outputs. Don't add anything extra and create your program to exactly match the specifications given above and in the samples. Your solution should be divided up into functions (your greeting should be a function too). No function body should have more than 15 lines of code. Comments don't count as line numbers.

Question: Can `main` have more than 15 lines?

Answer: No. Not for HW7. The header of a function does not count as a line number, just the code in the body of the function counts. Therefore, the `def main():` does not count as a line.

In the sample runs, the user input is shown in bold so you can easily distinguish input from program generated text (it does not have to show up as bold in your program).

You can use the `cat` command to display the contents of a file to the screen. The `cat` command is a linux tool, not a python command. Therefore, in the sample runs, the cat command is typed using the keyboard after the program was done running. Before running the program, I used emacs to create myFile.txt and placed it in the same folder as my hw7.py. The content of myFile.txt was:

```
Did you win $100?
YES!
```

Sample run 1

```
bash-4.1$ python hw7.py
This is a filter program that will convert your text file into ASCII values
Enter the name of the file: theFile.txt
Enter the name of the file: hello
Enter the name of the file: 2
Enter the name of the file: myFile.txt
Enter a base between 2 to 9: hello
Enter a base between 2 to 9: 2.0
Enter a base between 2 to 9: 2.5
Enter a base between 2 to 9: 10
Enter a base between 2 to 9: 1
Enter a base between 2 to 9: -8
Enter a base between 2 to 9: 8
Done.  Use cat to see the file:  8_myFile.txt
bash-4.1$ cat myFile.txt
Did you win $100?
YES!
bash-4.1$ cat 8_myFile.txt
104 151 144 .. 171
157 165 .. 167 151
156 .. .. 61 60
60 .. .. 131 105
123 .. ..
bash-4.1$
```

Sample run 2

```
bash-4.1$ python hw7.py
This is a filter program that will convert your text file into ASCII values
Enter the name of the file: myFile.txt
Enter a base between 2 to 9: 5
Done.  Use cat to see the file:  5_myFile.txt
bash-4.1$ cat myFile.txt
Did you win $100?
YES!
bash-4.1$ cat 5_myFile.txt
233 410 400 .. 441
421 432 .. 434 410
420 .. .. 144 143
143 .. .. 324 234
313 .. ..
bash-4.1$
```

Sample run 3 (assume a file called myOtherFile.txt has been created in the same folder as hw7.py. The content is shown after the program runs using the cat command)

```
bash-4.1$ python hw7.py
This is a filter program that will convert your text file into ASCII values
Enter the name of the file: myOtherFile.txt
Enter a base between 2 to 9: 4
Done.  Use cat to see the file:  4_myOtherFile.txt
bash-4.1$ cat myOtherFile.txt
num0
num1
num2
num3
num4
num5
num6
num7
num8
num9
bash-4.1$ cat 4_myOtherFile.txt
1232 1311 1231 300 ..
1232 1311 1231 301 ..
1232 1311 1231 302 ..
1232 1311 1231 303 ..
1232 1311 1231 310 ..
1232 1311 1231 311 ..
1232 1311 1231 312 ..
1232 1311 1231 313 ..
1232 1311 1231 320 ..
1232 1311 1231 321 ..

bash-4.1$
```

Question: I understand why all the `..` are there except for the very last one. Why does it seem like there is one extra `..` at the very end?

Answer: Emacs will place a new line (ASCII 10) at the end of the file even if you don't press the enter key.

IMPORTANT: Only submit hw7.py. Don't submit your sample input or output text files.

When you've finished your homework, use the submit command to submit the file.

You must be logged into your account and you must be in the same directory as the file you're trying to submit.

At the Linux prompt, type

```
submit cs201 HW7 hw7.py
```

After entering the submit command shown above, you should get a confirmation that submit worked correctly:

```
Submitting hw7.py...OK
```

If not, check your spelling and that you have included each of the required parts and try again.

You can check your submission by entering:

```
submitls cs201 HW7
```

You should see the name of the file that you just submitted, in this case hw7.py