**Number Bases**

Write a C/C++ program that does the following:

0. Using the division algorithm introduced in class to convert between base 10 and any other number base:

1. Prompts the user to enter an unsigned integer in base 10 (decimal) from the keyboard.

2. Prompts the user a new base ( greater than or equal to 2 and less than or equal to 36\*\* ) to convert the base 10 unsigned integer into

3. Constructs a string of digits that represents the user entered base 10 integer in the user entered new base

4. Output the constructed string

\*\* for bases greater than 10, you will want to follow the notation that hexadecimal does (10=A, 11=B, 12=C, 13=D, 14=E, 15=F ) meaning that a base 36 number system, the number 35 would be represented with Z, now if you want to go to base 62 (which is the 10 characters 0-9, upper case A-Z, and lower case a-z), you could have Z=35 (capital Z that is) and z=61 (lowercase Z).

Chart for digits up to base 36: (the first row is remainders in base 10 from our division algorithm, the second row is the digits of our base 36 number system. Hint: this works for all number systems up to base 36. For example hexadecimal (base 16) would be row 1 from 0 to 15 from its base 10, and its digits are 0-9, A-F)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |  | 32 | 33 | 34 | 35 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V |  | W | X | Y | Z |

\*\*\* I strongly encourage you to refine your program to as low level C/C++ (in simplest terms of statements) as you can possibly write it because we may be eventually writing this same program in Assembly Language!!!

This program will be due by 11:59PM on 9/16/2016. We will discuss this program in class on Wednesday 9/7/2016, but I am assigning it now so you can start to think about how to do this program (it may sound very challenging, but once we talk about a pseudocode in class, it should be really actually straightforward).

Here are some test cases for your program:

Unsigned Integer: 123 (in base 10)

New base: 2

Output string: 1111011

Unsigned Integer: 1234 (in base 10)

New base: 16 (hexadecimal)

Output string: 4D2

Unsigned Integer: 528 (in base 10)

New base: 6

Output string: 2240

Unsigned Integer: 8451 (in base 10)

New base: 36

Output string: 6IR

\* consult with the table above for base 36.

The division algorithm for this one is as follows:

1. 8451 / 36 = 234 with remainder of 27, 27 corresponds to R in the table, so the string ends with the R character

2. 234 / 36 = 6 with remainder of 18, 18 corresponds to I in the table, so the string is now IR

3. Since 6 is smaller than 36, our algorithm is done and we add 6 to the string, having: 6IR.

Now the math for checking our work: 6\*36^2 + 18\*36^1 + 27 = 7,776 + 648 + 27 = 8,451