CSCI 241 Assignment 1
100 points

Purpose

This assignment is an exercise in writing, compiling, and executing a C++ program on the departmental UNIX system. It also covers the creation and manipulation of C and C++ strings.

Set Up

1. Log in to UNIX.
2. Change to your `csci241` directory:

   ```
   cd csci241
   ```

3. Create a directory to hold your files for Assignment 1:

   ```
   mkdir Assign1
   ```

4. Change to your `Assign1` directory:

   ```
   cd Assign1
   ```

5. To create (or edit) a source code file, type:

   ```
   nano assign1a.cpp
   ```

   If the file `assign1a.cpp` does not exist, an empty text file will be opened for you. If it does exist, it will be opened for editing. You can type your source code into the file. To save, type Ctrl-o. To save and exit, type Ctrl-x.

6. To compile and link a program you've created, type:

   ```
   g++ -Wall -std=c++11 -o assign1a assign1a.cpp
   ```

7. To run the executable file created by the previous command, type:

   ```
   ./assign1a
   ```

   A makefile is not required for this assignment.

Overview

Numerology is the "study of the purported mystical or special relationship between a number and observed or perceived events." It has been used throughout human history as a way to attach meaning to a name, object or event using mathematics. It is considered a "pseudoscience" by modern scientists since it has no basis in observable phenomena. With that said, it's a good example of manipulating characters and strings, so we're

going to go with it!

One version of numerology ascribes meaning to the following numbers:

- 0 = emptiness, nothingness, blank
- 1 = independence, loneliness, creativity, originality, dominance, leadership, impatience
- 2 = quiet, passive, diplomatic, co-operation, comforting, soothing, intuitive, compromising, patient
- 3 = charming, outgoing, self expressive, extroverted, abundance, active, energetic, proud
- 4 = harmony, truth, justice, order discipline, practicality
- 5 = new directions, excitement, change, adventure
- 6 = love, harmony, perfection, marriage, tolerance, public service
- 7 = spirituality, completeness, isolation, introspection
- 8 = organization, business, commerce, new beginnings
- 9 = romantic, rebellious, determined, passionate, compassionate
- 11 = idealism, visionary, inspirational
- 12 = perfectionist, discriminating
- 22 = builder, leader, humanitarian, practical, honest

What you want to do for this project is to ask the user to type in their name. Next, you will need to use a technique called "theosophical reduction" to convert their name into a number. For example, the letter "a" is equal to the number 1. "b" = 2, "c" = 3, etc. Once you've gotten all of the letters converted into numbers you can add them up.

So for the name "craig" the numerology number would be:

c = 3
r = 18
a = 1
i = 9
g = 7

3 + 18 + 1 + 9 + 7 = 38

However, 38 is not on the "personality trait" lookup table above, so we need to further reduce the number by adding up its individual digits like so:

3 + 8 = 11

The number 11 **is** on the personality traits table, so we can print out for the user what the traits for the name they entered are based on this number.

Processing Requirements

For this assignment, you will write two versions of the numerology program. Name these two program files `assign1a.cpp` and `assign1b.cpp`. The first version of the program will use objects of the C++ `string` class to store strings. The second version will use C strings (null-terminated arrays of `char`) to store strings.

The logic for both versions of the program is similar:

1. Prompt the user to enter a name.
2. Read the string the user types. Keep in mind that a name may contain a non-alphabetic character such

as a hyphen, apostrophe, or even a space.
3. Loop through the characters of the string, converting them to numbers and adding the numbers to a sum.
4. If the sum is equal to one of the numbers in the "personality trait" table above, you've found the traits for this name. If not, you'll need to further reduce the number as follows:
    A. Convert your sum to a new string.
    B. Loop through the characters of this new string, converting them to numbers and adding the numbers to a sum.
    C. Go back to Step #4.
5. Print the traits for the name.
6. Ask the user whether or not they want to enter another name, and read their response. If the response is "y" or "Y", go back to Step #1.

Some useful string-manipulation techniques for this assignment are described below.

Reading a string that contains whitespace

While the >> operator can be used to read a string of characters into either a C++ `string` object or an array of `char`, it will not work correctly if the string entered by the user contains whitespace (a space or tab), because it stops reading characters the moment it hits whitespace.

The `getline()` function in the header file `<string>` can be used to read a string that contains spaces or tabs into a C++ `string` object.

The `getline()` method of the `istream` class in the header file `<iostream>` can be used to read a string that contains spaces or tabs into an array of `char`.

Converting the characters of a string to numbers

Write a function to convert a character to a number. This function should take a `char` argument and return an `int`. Here's some possible logic for the function:

- If the character is a letter, return the corresponding number from 1 to 26 ('A' or 'a' = 1, 'B' or 'b' = 2, 'C' or 'c' = 3, etc.).
- If the character is a digit, return the corresponding number from 0 to 9 ('0' = 0, '1' = 1, '2' = 2, etc.).
- Otherwise, return 0 (so that this character has no effect on the sum).

Remember that characters in C++ are actually stored as small integers. That means that you can easily convert a letter or digit character to an integer simply by performing some basic math. A big cascading `if` or `switch` statement is **not** the best way to do the conversion!

You can use the character functions found in the C standard library header file `<cctype>` to determine whether a character is a letter or a digit, and to convert all of the letters to the same case.

Converting numbers to strings

This page has some discussion of converting strings to numbers and numbers to strings, including example code for both C++ strings and C strings.

Sample Output

Both versions of the program should produce the same output. Here's a sample run of the program with several names used as input:

```
Numerology Program

Enter a name: Kurt
That name has the traits spirituality, completeness, isolation, introspection

Another? (y/n) y

Enter a name: Amy
That name has the traits perfectionist, discriminating

Another? (y/n) Y

Enter a name: Joe Bob
That name has the traits harmony, truth, justice, order discipline, practicality

Another? (y/n) y

Enter a name: Xiao-Li
That name has the traits spirituality, completeness, isolation, introspection

Another? (y/n) y

Enter a name: Prasad
That name has the traits new directions, excitement, change, adventure

Another? (y/n) n
```

Other Points

- As always, programs that do not compile on `turing/hopper` automatically receive 0 points.
- You can submit both versions of your program as a single archive file by using the electronic submission guidelines posted on the course web site and described in class.