```
Script started on Thu 19 Apr 2012 12:08:41 AM CDT
\033]0;georgia@georgia-MT6017: ˜/cplusplus\007georgia@georgia-MT6017:˜/cplusplus$ p
wd
/home/georgia/cplusplus
\033]0;georgia@georgia-MT6017: ˜/cplusplus\007georgia@georgia-MT6017:˜/cplusplus$ c
at suffixez.info
Name: Jakob Hansen
Class: CSC121 - Evening Section
Lab: For the nth time

Levels attempted:

2 - For just getting the basic instructions down.

+2 for putting the random message generator in a function.


Program Description:

A program that will append the appropriate suffix to ordinal numbers. The user
enters a number, and the program prints the number with the correct suffix, and
with a witty, dynamic message to boot!








\033]0;georgia@georgia-MT6017: ˜/cplusplus\007georgia@georgia-MT6017:˜/cplusplus$ c
at suffixez\033[K.cpp
#include <iostream>
#include <climits>
#include <string>
#include <cctype>
#include <cstdlib>

using namespace std;

string rand_mess_gen(short choice);

int main(void)
{
    long number;
    short ones_place, tens_place, num;
    char yes_no;
    string suffix;

    srand(time(NULL));

    cout << "\nWelcome to the ordinal number suffix program!\n"
            "Would you like to enter a number?  ";

    cin >> yes_no;

    while (toupper(yes_no) == ’Y’)
    {
        cout << "\n\nPlease enter a number:  ";

        cin >> number;

        ones_place = number % 10;

        tens_place = number/10 % 10;

        if (tens_place == 1)
        {
            suffix = "th";
        }
        else if (ones_place == 1)
        {
            suffix = "st";
        }
        else if (ones_place == 2)
        {
            suffix = "nd";
        }
        else if (ones_place == 3)
        {
            suffix = "rd";
        }
        else
        {
            suffix = "th";
        }

        num = rand() % 5 + 1;


        cout << rand_mess_gen(num) << number << suffix << "\n\n"
                "Would you like to enter another number?  ";

        cin >> yes_no;

        cin.ignore(INT_MAX, ’\n’);

    }

    cout << "Thanks, Bye!\n\n";




    return 0;
}

string rand_mess_gen(short choice)
{
    string message;

    switch(choice)
    {
    case 1:
        message = "\nThe answer you’re looking for is ";
        break;
    case 2:
        message = "\nThe the correct suffix appended is ";
        break;
    case 3:
        message = "\nI believe that would be ";
        break;
    case 4:
        message = "\nSomething tells me that it’s ";
        break;
    case 5:
        message = "\nI couldn’t possibly think of another way to say ";
        break;
    }

    return message;
}
\033]0;georgia@georgia-MT6017: ˜/cplusplus\007georgia@georgia-MT6017:˜/cplusplus$ C
PP suffixez
suffixez.cpp***
suffixez.cpp: In function â\200\230int main()â\200\231:
suffixez.cpp:31:31: warning: conversion to â\200\230short intâ\200\231 from
â\200\230long intâ\200\231 may alter its value [-Wconversion]
```

```
suffixez.cpp:33:34: warning: conversion to â\200\230short intâ\200\231 from
â\200\230long intâ\200\231 may alter its value [-Wconversion]
suffixez.cpp:56:28: warning: conversion to â\200\230short intâ\200\231 from
â\200\230intâ\200\231 may alter its value [-Wconversion]


\033]0;georgia@georgia-MT6017: ˜/cplusplus\007georgia@georgia-MT6017:˜/cplusplus$ .
/suffi\033[Kxez.out

Welcome to the ordinal number suffix program!
Would you like to enter a number?  y


Please enter a number:  21

The the correct suffix appended is 21st

Would you like to enter another number?  y


Please enter a number:  32

I couldn’t possibly think of another way to say 32nd

Would you like to enter another number?  y


Please enter a number:  43

The the correct suffix appended is 43rd

Would you like to enter another number?  y


Please enter a number:  11

Something tells me that it’s 11th

Would you like to enter another number?  y


Please enter a number:  219

Something tells me that it’s 219th

Would you like to enter another number?  n
Thanks, Bye!

\033]0;georgia@georgia-MT6017: ˜/cplusplus\007georgia@georgia-MT6017:˜/cplusplus$ c
at suffixez.tpq
1. What two digits of the integer determine which suffix it should have? (Hint:
One is typical and one is special/exceptional...)

    The ones place and the tens place digits determine the suffix. The
    exceptional digit is the tens place, because the only time it plays
    any role at all in determining the suffix is when it is 1.

2 .How do you extract these digits from the user’s whole number? (Yes, it is
supposed to be an integer -- not a string! Note the integer math in the topic
list above...)

    Using the MODULO operator. In a base 10 system, the ones place is extracted
    by modulo-ing the number that you are extracting from by 10. In numbers
    larger than 10, you have to move the decimal point over as many places as
    you need. Luckily, integer division will chop off messy decimals, allowing
    us to move the decimal point by dividing our number by the place we wish to
    isolate ie. 347 / 10 = 34. once we’ve moved our the appropriate digit into
    the ones place, we just mod by 10 again, and we’ve extracted our digit.
```

```
3. How many branches does it take to differentiate the five cases from one
another? ("One case for the exceptional rule; three cases for the regular rules;
and one case for everyone else!") How many conditions must be tested in this
branching? Are any of these branches cascaded from/on/with one another?

    There are five branches - one for each case. Four conditions need to be
    tested with if or else if statements. All of the branches are part of the
    same if/else if statement.

4 + 5 How can your program allow the user to type both ’y’ and ’yes’ for their
again response? (Hint: This part of the program does NOT involve strings!!!)
How can your program allow the user to type both ’y’ and ’Y’ for their again
response? (Hint: There are two ways, but one is far easier...)

    To allow both ’y’ and ’yes’ you use cin.ignore(INT_MAX, ’\n’) to just read
    the first char of the input and throw everything else away.

    To allow both ’y’ and ’Y’ you use the toupper(char) function from cctype
    to test only the uppercase version, eliminating the need to test twice as
    many conditions.

6. Are there any loops besides your yes/no loop? (Hint: there needn’t/shouldn’t
be...)

     No other loops

How many tests would be needed to thoroughly test your suffix generation
algorithm? (You can ignore the yes/no loop around it for this calculation...)

    You need to test each possible relevant condition:

    three numbers with a 1, a 2, and a 3 in the ones place but which do NOT have
    a one in the tens place

    ANY number with a 1 in the tens place and 1, 2, or 3, in the ones place.

    ANY number that has a 5, 6, 7, 8, or 9 in the ones place.

    So...5.


BTW, how can you have your program print different response text before the
suffix’ed number result? (You did notice that in the sample run above, didn’t
you? You’ve got a good eye for detail!) (Hint: Think of it as a random message.

    I used a switch statement where each branch sets a string to a different
    message. I put this in a function to which I pass a random value and which
    returns the random string that gets "chosen".




\033]0;georgia@georgia-MT6017: ˜/cplusplus\007georgia@georgia-MT6017:˜/cplusplus$ e
xit
exit

Script done on Thu 19 Apr 2012 12:11:11 AM CDT
```