Angie Chen

GE1501 18472 SEC 10

Professor Hertz

2 October 2016

Assignment 4 Report: Expanding Exponentially like some Recursive Virus

To accommodate the band’s crowd’s complex growing or decreasing proportion of fans, a program based on an equation and a given R constant, estimations can be made about the band’s success or failure. The given equation (below) representing the population mentality, where represents the popularity in the current week and represents the popularity of the week before, accommodates two phenomena: growth of popularity “in proportion to the number of people who are currently fans”, and the loss of some fans who have an aversion to the band’s increasing popularity (2 Hertz).

The first week’s percentage of popularity (), and three sample R values (0.90, 1.50, 3.95) were given. To truly understand the outcome of the band’s success or failure, weekly outputs reflecting the band’s weekly changing popularity are displayed.

As usual, I brainstormed the general sequence of logic this program should follow. The idea of this program reminded me of how a program producing the Fibonacci sequence ran (shown in Appendix A and Fig. #) (ProgrammingSimplified). Following this structure, I composed my program after writing a series of pseudocode displayed below. In addition to pseudocode, the isolation and checking of the functionality of the equation itself is shown in Appendix B.

1-2 Include iostream, library for more complex arithmetic

4 Maintain standard versions of functions/variables

5 Function automatically applied to every line of code below

8 Ensuring that the all of the variables can store decimals

10-11 Asking the user for an R input to store

12 Display the answer sentence for proper run order

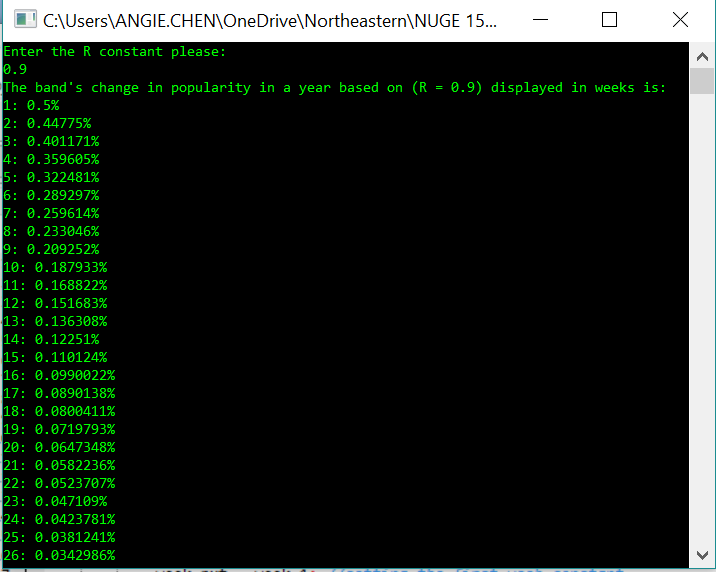
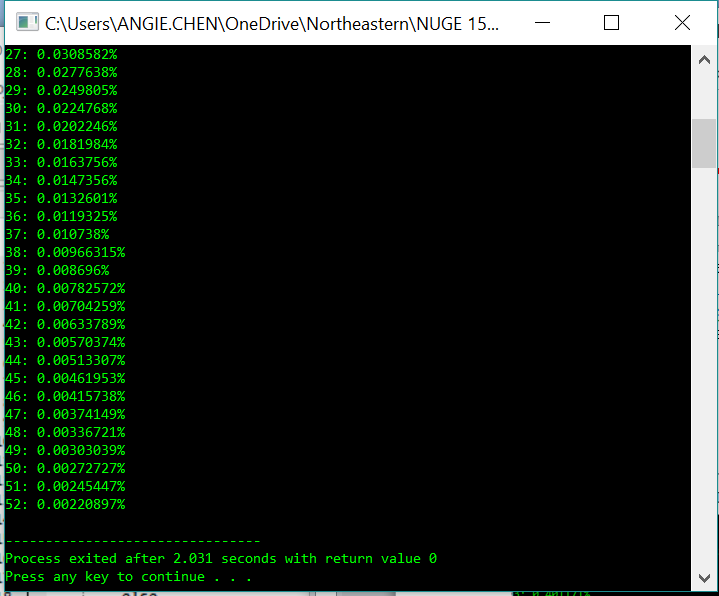
14 Initializing a more compact form of do/while, and increasing the week number by 1 each cycle up to 52 cycles

15-17 Declaring and defining the value for the first week

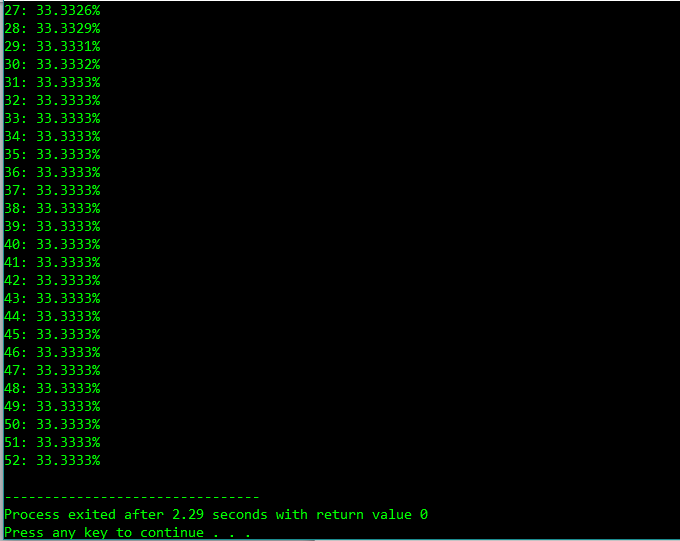
18-23 Calculating the value for the subsequent week and declaring the value as the initial value for the next cycle

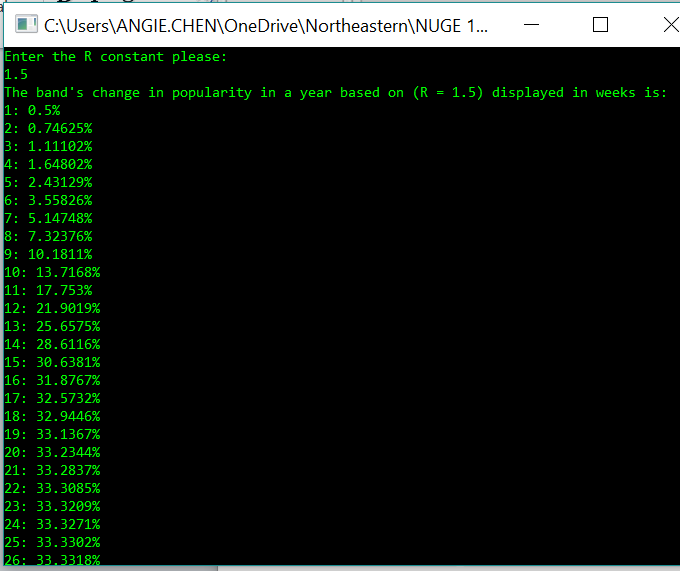
27 End program

Below are screenshots and analysis of the success of the band predicted using the given R constants. Appendix B contains the code for the program.

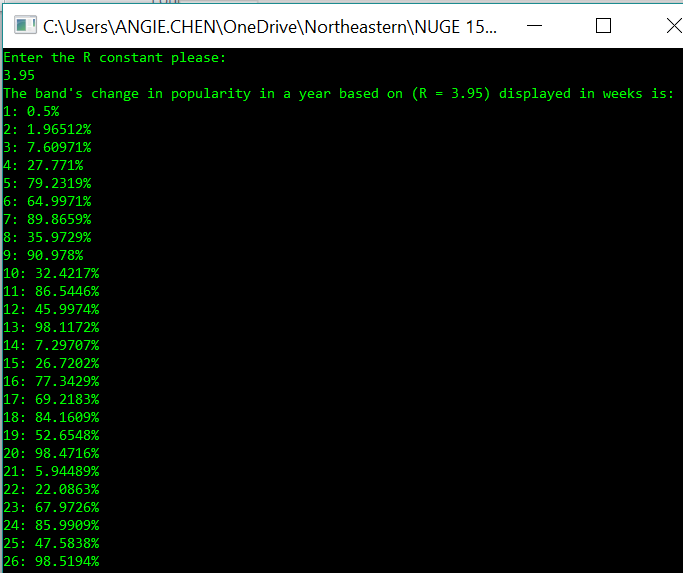
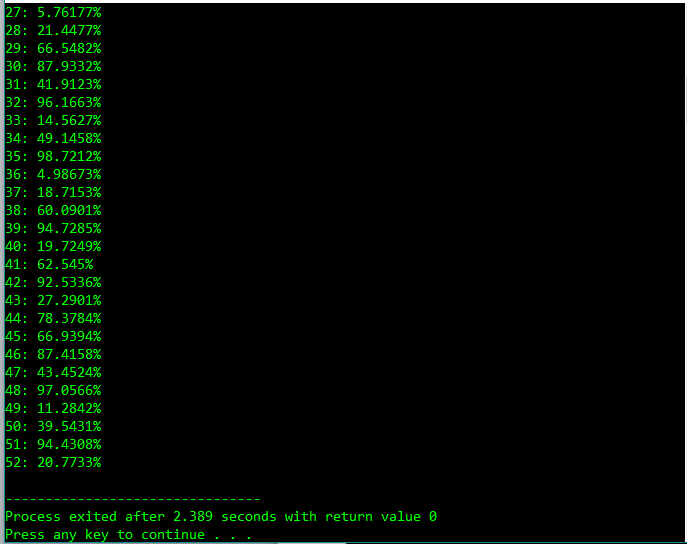


When R is defined as 0.9, it can be seen that the band ends up steadily losing popularity.





When R is defined as 0.9, it can be seen that the band’s popularity evens out at about 33%.



When R is defined as 0.9, the band’s change in popularity seems random, with an even amount of peaks of successes and dips of extreme uninterest.

**Works Cited**

Hertz, Joshua. “A4: Expanding Exponentially like some recursive virus”. Blackboard.neu.edu. 2016. Web. 30 September 2016.

Programming Simplified. “C++ program to generate Fibonacci series”. Programming Simplified: Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License. Web. 1 October 2016.

**Appendix A**

#include <cstdlib>

using namespace std;

int main()

{

int n, c, first = 0, second = 1, next;

cout << "Enter the number of terms of Fibonacci series you want" << endl;

cin >> n;

cout << "The first " << n << " terms of the Fibonacci series are : " << endl;

for ( c = 1; c <= n; c++)

{

if ( c <= 1 )

next = c;

else

{

next = first + second;

first = second;

second = next;

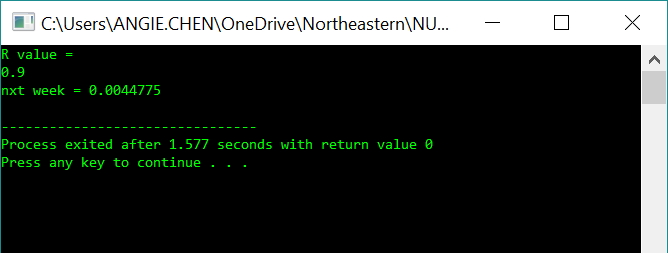
}

cout << c << ": " << next << endl;

}

return 0;

}



The output value is equivalent to that of true program where R = 0.90.

**Appendix B**

using namespace std;

int main()

{

float R, week\_1, week\_nxt;

cout << "R value = " << endl;

cin >> R;

week\_1 = 0.005;

week\_nxt = (R \* week\_1 \* (1-week\_1));

cout << "nxt week = " << week\_nxt << endl;

return 0;

}

**Appendix C**

#include <iostream>

#include <cstdlib>

using namespace std;

int main()

{

float R, week\_1 = 0.005, week\_nxt, week\_num; //ensuring that the code can account for decimals

cout << "Enter the R constant please: " << endl;

cin >> R;

cout << "The band's change in popularity in a year based on (R = " << R << ") displayed in weeks is: " << endl;

for(week\_num = 1; week\_num <= 52; week\_num ++) //increasing the week number by 1

{

if( week\_num <= 1)

week\_nxt = week\_1; //setting the first week constant

else

{

week\_nxt = R \* week\_1 \* (1 - week\_1); //defining the equation for the change in popularity

week\_1 = week\_nxt; //declaring the new value for "week\_1" in the next cycle

}

cout << week\_num << ": " << week\_nxt \* 100 << "%" << endl; //printing the week number and the popularity percentage

}

return 0;

}