Virtual Civilization

New Mexico Adventures in Supercomputing Challenge
Final Report
April 2, 2002

Team_047
Manzano High School

Team Members
Bryan Dean
Sean Gentry
Amjad Musleh

Teachers
Stephen Schum

Project Mentor

<u>Table Of Contents</u>

Executive Summary

Human populations grow and shrink according to several well-studied rules, such as population ceilings and demographic transition. However, these rules have not been applied to a real model of a population. One reason for this is that the changes take place over a long period of time and finding the exact population of a large group of people is difficult. With a computer model, it is possible to not only observe a large population over a long period of time, but also allows variables such as starting population and population ceilings to be changed. The relative technological level of a population could also be calculated.

This computer model would also have to factor in war as it impacts population. War would have to be a random element as war cannot be predicted over the long term. It would also have to increase in damage as the technological level increases, as new technology often brings more destructive weapons.

The purpose of this project is to simulate the growth of a human population, factoring in war and technological advancement. The results can then be used to find the optimal population ceiling and starting population of a civilization.

Introduction

The world we live in today is a complex one. Technologies, natural resources, disasters, population growth…all of these are factors in our current society. But can these effects be properly illustrated in a computer simulation? Is there a mathematical model that can quantify the effects of these variables? The purpose of this project is to create several differently sized simulated populations and trace the growth of each until they die out. We will factor in available resources, technological development, conflict and disease, and other miscellaneous effects on the total population.

The results of this simulation can be a baseline for charting the timeline of an artificial colony or nation. By taking a certain size population and inputting that into the model, we can observe and record the survival rate of the population and adjust factors, such as original population and available resources, accordingly.

Description

Our group has learned many important skills required for success in the Adventures in Supercomputing Challenge. We spent three weeks learning the general aspects of html. We spent two to three weeks setting up both MHS AiS Challenge Linux accounts and accessing our New Mexico AiS Challenge Unix accounts. We discussed and researching general math algorithms for computing distances and velocities in simple motion problems, volumes of rectangular and cylindrical railroad cars, and creating mailing record databases. We spent five weeks learning the basic concepts of

C++ programming code. Combining the math algorithms with programming skills, we then wrote the C++ code to compute the volume of rectangular and cylindrical railroad cars, solving physics problems of motion, and creating a mailing record database. Basically, our instructor provided us with nominal information and starting templates to write the program code for the above sample problems.

The purpose of this project is to create several differently sized simulated populations and trace the growth of each until they die out. We will factor in available resources, technological development, conflict and disease, and other miscellaneous effects on the total population. To begin, researched the average consumption of resources per person, the rate of regeneration of these resources, the frequency of war and disease, and technology's effect on a culture. These will be put into a computer model using C++ coding techniques. A graphic representation of the population growth may be included as well.

Results

We created our own variables for our project to help calculate the data easier. Some of the variables are defined as the following.

1. **T = Technology Level of the population. T is calculated by dividing the current year by 1000.**
2. **High Point = Highest point the population grows to.**
3. **Resources = Resources the population uses and limit the population can grow to. The default calculation for Resources is calculated (pow(10,9)) or 10 to the 9^{th} power.**

4. **Birth Rate = The number of births the population has in a year. The Birth Rate Decreases .001 every year after (T<7.5 & T>5). The Starting is Birth Rate = .92**
5. **Death Rate = The number of Deaths the population has in a year. The Death Rate Decreases .001 every year after (T<7.5 & T>5). The Starting is Death Rate = .91568**
6. **Current Pop = The current population at the end of the year. The current population = (current pop+((birth_rate-death_rate)*current pop)*((resources-current_pop)/resources))**
7. **War Pos = The possibility of war. The war is a random event in the program so it is calculated, war_pos=(rand()%100)+1**

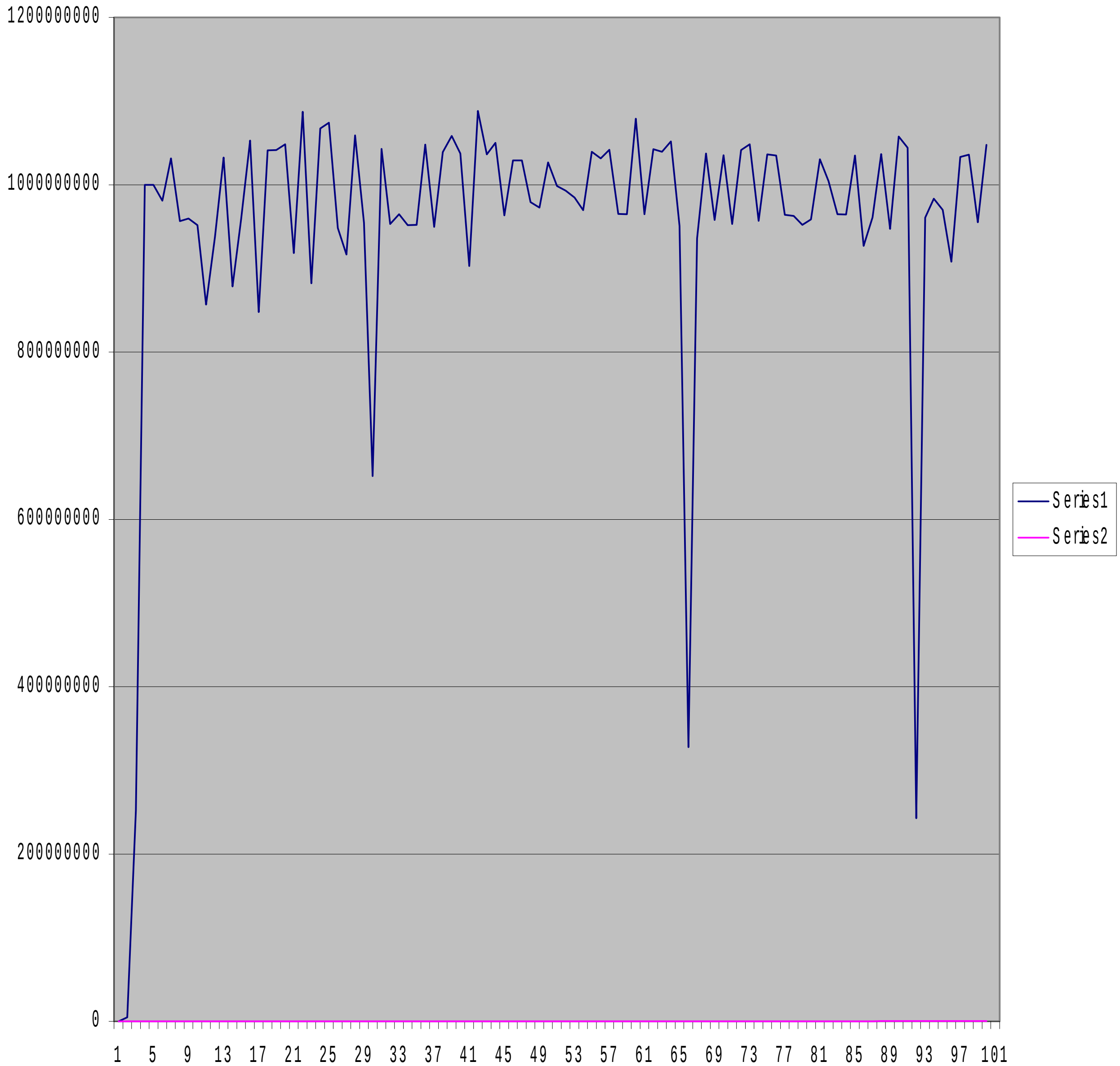
Conclusion

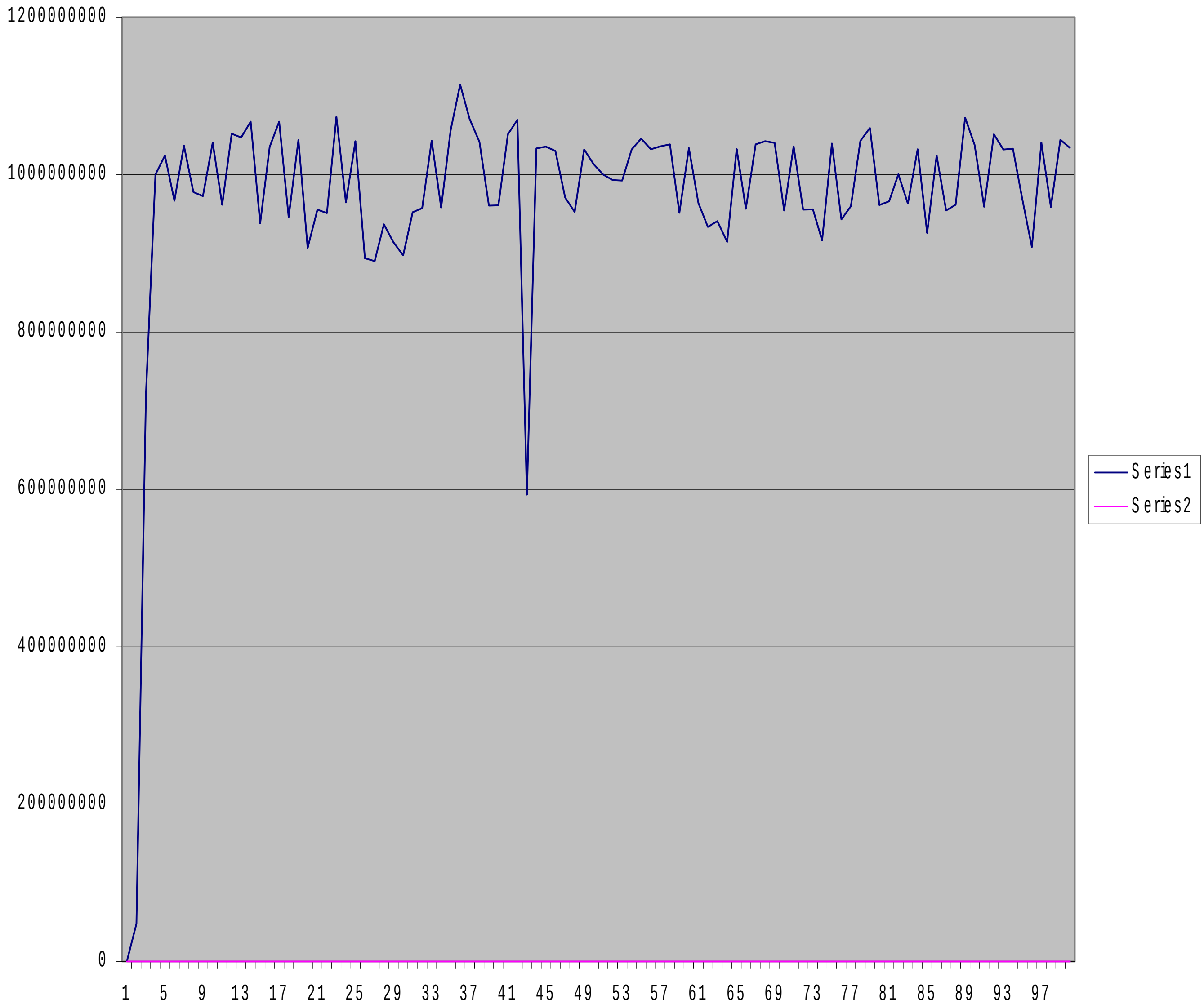
The results indicate that, using our model, a civilization of any starting size will die out around the year 100000. This is not consistent with our original hypothesis in that the starting size of a population seems to have little effect on the end result. Also, the war algorithm seems too simple. With more research, perhaps a better algorithm for war could have been developed. Despite these problems, we have nevertheless created an accurate model of demographic transition. The birth and death rates change accordingly and the population, once it approaches the population ceiling, begins to stabilize. Perhaps with a more powerful computer, several simulations could run and even interact at the same time.

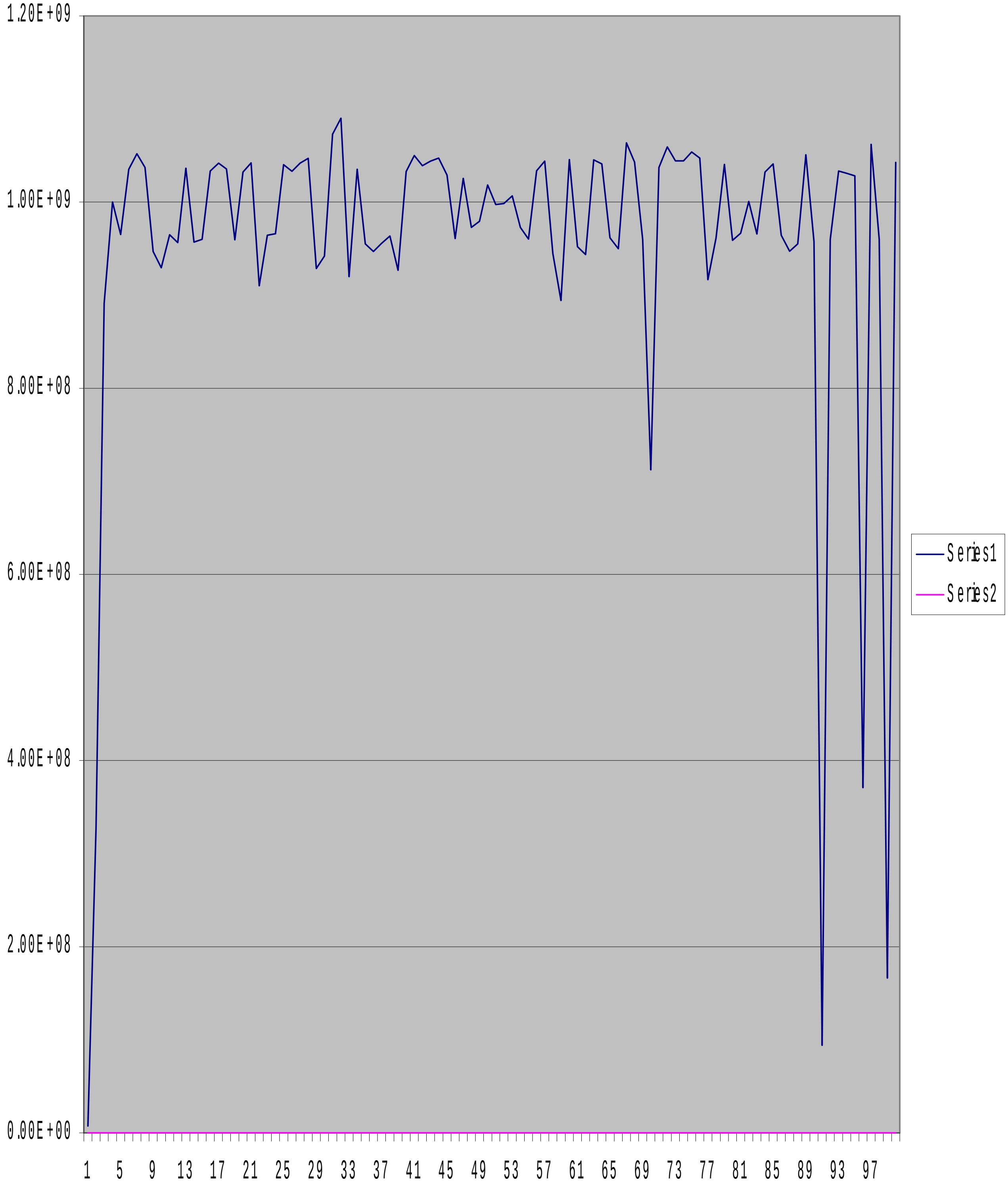
Bibliography

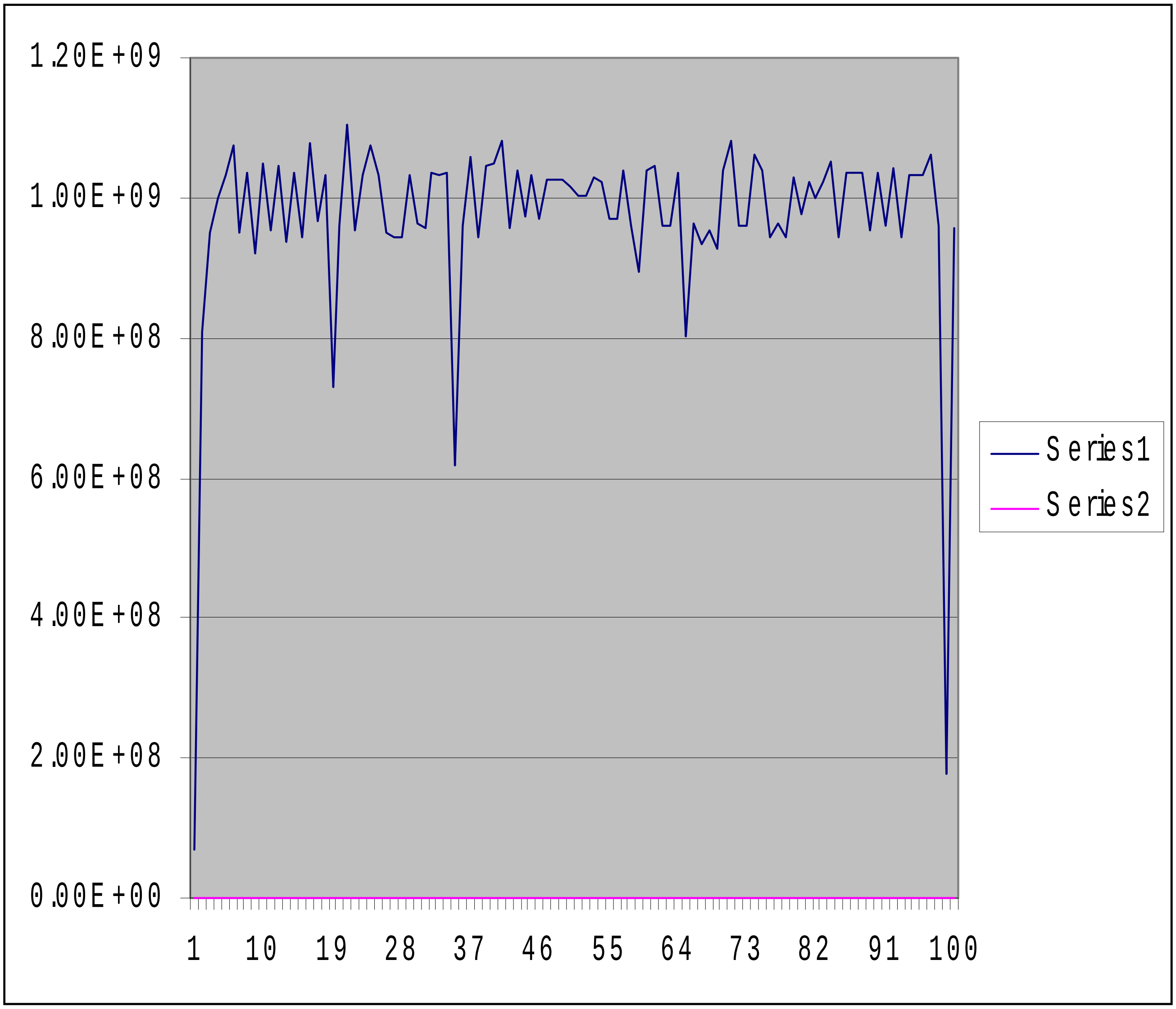
1. Eric Strauss. Biology the Web Of Life, Scott Foresman-Addison Wesley: Menlo Park, CA, 1998
2. DR. K. M. SLOMCZYNSKI. "SOCIAL STRATIFICATION IN HUMAN SOCIETIES: THE HISTORY OF INEQUALITY" < http://www.soc.sbs.ohio-state.edu/classes/Soc463/slomczynski/Ch03.htm>
3. Kent Reisdorph. Teach Yourself C++ in 5 Days, Sams Publishing: Indianapolis, 1999
4. Joel Adams. C++: An Introduction to Computing, 2nd Edition. Prentice Hall: Upper Saddle River, New Jersey, 1998
5. Barbara Johnston. C++ Programming Today. Prentice Hall: Upper Saddle River, New Jersey, 2002

LIST OF APPENDICES


1200000000
1000000000
800000000
600000000
400000000
200000000
0
1
5
9
13
17
21
25
29
33
37
41
45
49
53
57
61
65
69
73
77
81
85
89
93
97
101
Series1
Series2


1200000000
1000000000
800000000
600000000
400000000
200000000
0
1
5
9
13
17
21
25
29
33
37
41
45
49
53
57
61
65
69
73
77
81
85
89
93
97
Series1
Series2


1.20E+09
1.00E+09
8.00E+08
6.00E+08
4.00E+08
2.00E+08
0.00E+00
1 5 9 13 17 21 25 29 33 37 41 45 49 53 57 61 65 69 73 77 81 85 89 93 97
Series1
Series2



1.20E+09
1.00E+09
8.00E+08
6.00E+08
4.00E+08
2.00E+08
0.00E+00
1
10
19
28
37
46
55
64
73
82
91
100
Series1
Series2

```
* virtciv6.cc  Bryan Dean Sean Gentry Amjad Musleh       4/2/03
This is a logistical population model based on the variables in our program.
It factors in population limits and random wars, as wars cannot be
predicted through an algorithm. */
#include <stdio.h>
#include <stdlib.h>
#include <iostream.h>
#include <math.h>
#include <fstream.h>  //needed for file output
#include <iomanip.h>  //needed for file output
#include <time.h>     //needed for rand() function
//Base Values

double start_pop=100;          //Starting population (Default)
int Y=1;                        //Basic unit of time (one year)
double start_res=(pow(10,9)); //Maximum population allowed by
environments

void main(){

  cout.precision(0);
  cout.setf(ios::showpoint|ios::fixed); //set output format

  double T=0,high_point=0,resources=start_res, current_pop=start_pop,
birth_rate=.92,   death_rate=.91568, temp;
  int war_num=6, year=Y;  //variables
  srand(time(NULL));      //seeds random number generator
  int ch=0;

  char buff[81];
  ofstream outfile("test.csv"); //creates spreadsheet file
  if (!outfile) return 0;

  cout<< "\033[2J"; //clears screen
  cout<<"This program will compute the population of a civilization."
<<endl<<endl<<"Use default values? (1 is 'yes, 2 is 'no'): ";
  cin>>ch;
  if (ch == 2){          //allows user-defined parameters
    cout<<"Enter starting population: ";
    cin >>current_pop;

    if (current_pop<2){
      cout<<"Invalid value: Population must be greater than 2."<<endl;
      exit(0);
    }    //quits program if unusable values are given
    cout<<"Enter population limit (10^x): ";
```

```
    cin>>temp;
    if (temp<1){
      cout<<"Invalid value: Limit cannot be less than 1."<<endl;
      exit(0);
    }
    resources=(pow(10,temp));
  }

  while (current_pop>0){
    if (year%1000==0) cout << "Current population: " << current_pop<<"
"<<"Year:"<<year<<endl;
    current_pop=(current_pop+((birth_rate-
death_rate)*current_pop)*((resources-current_pop)/resources)); //logistic
model

    T=year/1000;      //technology level
    int war_pos=(rand()%100)+1;          //random number for war
    if (war_pos==war_num){
       current_pop-=(current_pop*(T/100)); //formula for war casualties
    }
    if (T<5 && T>2.5){
      death_rate-=.001;

    } else if (T<7.5 && T>5){
          birth_rate-=.001;
       death_rate-=.001;

    } else if (T<10 && T>7.5){
          birth_rate-=.001;
       death_rate-=.001;

    }

    if (current_pop>high_point){
      high_point=current_pop;
    }

    if (year%1000==0) outfile << current_pop << "," <<year<< endl; //write file
    ++year;
  }
cout<<endl<< "Maximum population: "<<high_point<<endl<<"Last year:
"<<year<<endl;
outfile.close;
}
```