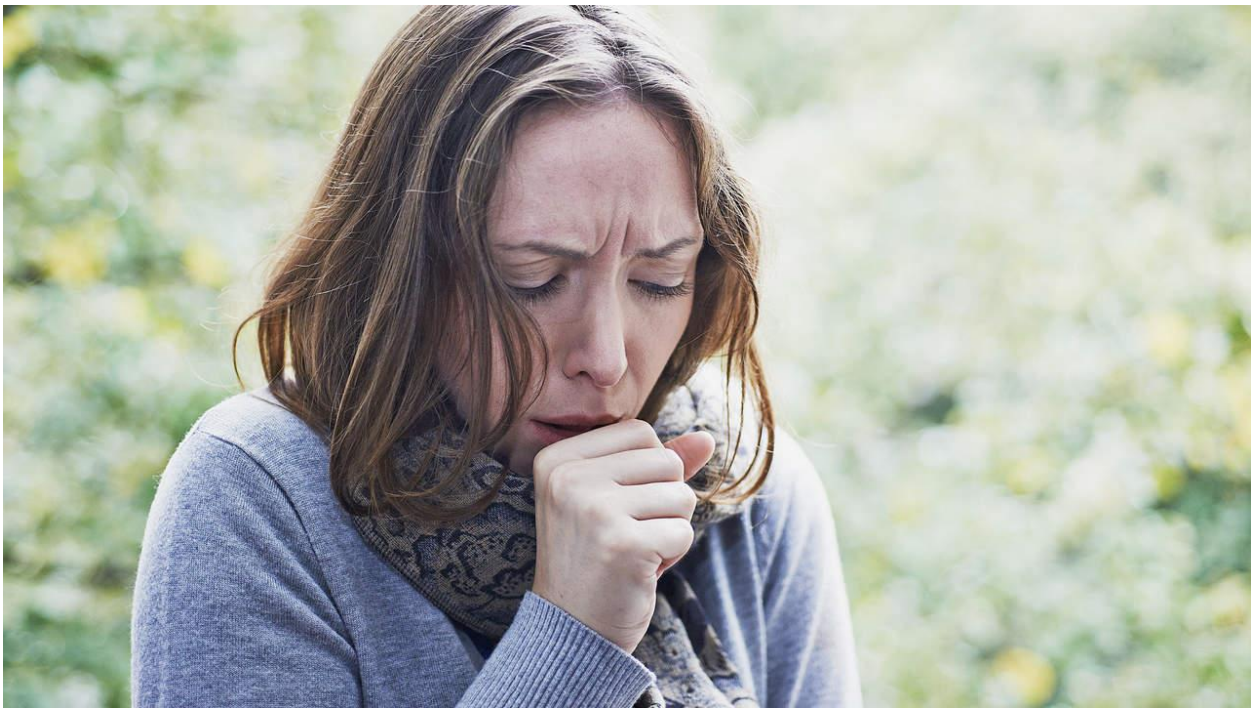
Daniel Espinoza

SDS 322 Introduction to Scientific Programming

Charlie Dey and Carrie Arnold

Final project



Main.cc

```
#include <iostream>
#include <ctime>
#include <cstdlib>
#include "person.h"
using namespace std;
int main()
{
    Person joe;
    srand(time(NULL));
    int step=0;
// updates joe
    joe.update();
// this while loop will continue until recovery
    while(joe.is_stable()==false)
    {
        // day counter
        step=step+1;
// updates every day
        joe.update();
// each day the person gets assigned bad luck to see if will get sick
        float bad_luck =(float) rand()/ (float) RAND_MAX;
        if(bad_luck>.95)
            {
//if his bad luck is high enough he will get sick
            joe.infect(5);
            }
// prints the status of the person of each day
    cout<<"On day "<<step<<", Joe is "<< joe.status_string()<<endl;
    if(joe.is_stable()==true)
    {break;}
    }
return 0;}
```

#include <ctime>

Uses the clock on my computer, so when I ask for a random number it will give us a new number based on the time, this allow us to infect Joe a different day each time instead of infecting Joe the same day every time we run the program.

#include <cstdlib>: Lets us use functions like srand().

#include <person.h>: Uses the methods in my person header file

Time(NULL): Resets the time to zero.

Person joe: joe is created as a Person

Joe.update(): Updates the status of joe.

Joe.is_stable(): checks if joe has recovered.

Joe.infect (int): infects the person joe for n days

Joe.status_string() : prints out the status of joe.

On day 1, Joe is susceptible
On day 2, Joe is susceptible
On day 3, Joe is susceptible
On day 4, Joe is susceptible
On day 5, Joe is susceptible
On day 6, Joe is susceptible
On day 7, Joe is susceptible
On day 8, Joe is susceptible
On day 9, Joe is sick
On day 10, Joe is sick
On day 11, Joe is sick
On day 12, Joe is sick
On day 13, Joe is sick
On day 14, Joe is sick
On day 15, Joe is recovered

On day 1, Joe is susceptible
On day 2, Joe is susceptible
On day 3, Joe is susceptible
On day 4, Joe is susceptible
On day 5, Joe is susceptible
On day 6, Joe is susceptible
On day 7, Joe is susceptible
On day 8, Joe is susceptible
On day 9, Joe is susceptible
On day 10, Joe is susceptible
On day 11, Joe is sick
On day 12, Joe is sick
On day 13, Joe is sick
On day 14, Joe is sick
On day 15, Joe is sick
On day 16, Joe is recovered

```
// Main file
#include <iostream>
#include <vector>
#include "person.h"
#include "population1.h"

using namespace std;
int main(){
// Initialize and set the amount
int npeople;
cout<<"How many people"<<endl;
cin>>npeople;
// Creates the population with npeople as the input
Population population(npeople);
// Infects a random person
population.random_infection();
int step=0;
while(population.count_infected()>=0)
{  // count
      step+=1;
      cout<< "In step   "<<step<<" ";
      // prints out the results
      population.Printer();
      // updates the population
      population.update();
      // if there is no more sick people then loop is broken
      if (population.count_infected()==0)
      {break;}
};
// adds an extra step since the loop when no one is sick
// and does not print the last line
cout<<"In step   "<<step+1<<" ";
population.Printer();
cout<<"Disease ran its course by step "<<step+1<<endl;
return 0;
}
```

#include<vector>

Lets us create and use vectors in our program.

#include “person.h”

Uses the person header file and its methods

#include “population1.h”

Uses the population1 header file and its methods.

Population population(neople):

Creates a population with npeople as the size, uses the Population class and its members.

Random_infection(): randomly infects a person

Count_infected(): counts the infected of that day

Printer(): out a line of statuses.

Update(): updates each member of the population using the update method in the original person class.

Main1.cc purpose.

The population is made of “Person”s and every time it loops it goes through each person of the population until all are safely recovered or healthy. Last part simply prints out another status line since the loop ends before it can print the last line of recovery. But since only one person get infected this time, it will take six days.

```
How many people

20

In step   1 #sick: 1:??+??????????????????

In step   2 #sick: 1:??+??????????????????

In step   3 #sick: 1:??+??????????????????

In step   4 #sick: 1:??+??????????????????

In step   5 #sick: 1:??+??????????????????

In step   6 #sick: 0:??-??????????????????
```

```
How many people

30

In step   1 #sick: 1:????????+????????????????????????

In step   2 #sick: 1:????????+????????????????????????

In step   3 #sick: 1:????????+????????????????????????

In step   4 #sick: 1:????????+????????????????????????

In step   5 #sick: 1:????????+????????????????????????

In step   6 #sick: 0:????????-????????????????????????
```

```
#include <iostream>
#include <vector>
#include "person.h"
#include "population.h"
#include <ctime>
#include <cstdlib>
#include <iomanip>
using namespace std;
int main(){
// Initializes and reads in the values needed
int npeople;
float probs;
cout<<"How many people"<<endl;
cin>>npeople;
cout<<"Infection chance ?"<<endl;
cin>>probs;
// Creates the population with Population class.
Population population(npeople,probs);
population.random_infection();
int step=0;
srand(time(NULL));
while(population.count_infected()!=0)
{
      step+=1;
      cout<< "In step   "<<setw(3)<<step<<" ";
      population.Printer();
// Every iteration it creates a new random float value
      float badluck=(float) rand() / (float) RAND_MAX;
//     Updates the status of the people of the population
      population.update();
//     Inputs the number into the setprob method
//     if the badluck is less than the spread rate then they will get sick
      population.setprob(badluck);
      };
cout<<"In step   "<<setw(3)<<step+1<<" ";
population.Printer();
cout<<"Disease ran its course by step "<<step+1<<endl;
cout<< population.sickcount()<<" got sick"<<endl;
return 0;
}
```

```
How many people
20
Infection chance ?
.5
In step    1 #sick:   1:???+???????????????
In step    2 #sick:   3:??+++???????????????
In step    3 #sick:   3:??+++???????????????
In step    4 #sick:   3:??+++???????????????
In step    5 #sick:   3:??+++???????????????
In step    6 #sick:   2:??+-+???????????????
In step    7 #sick:   0:??---???????????????
Disease ran its course by step 7
3 got sick
```

#include <iomanip>

Lets us manipulate the printing to help to keep the output uniform.

“population.h”

Uses the population header file and its methods.

Setprob(badluck)

If the random badluck assigned to the individual is less than then infection chance then they will get sick

Sickcount()

Counts the total amount who got infected and recovered.

Main2.cc purpose.

First, we read in the size and infection chance. Then we create a population with those parameters. We then randomly infect a person in this population. The program loops until there is no more sick people in the population. The loop continuously updates and prints out the statuses of the persons in the population. After updating it sees if the left and right will get infected, generating a new float value every time it loops. Once there is no current sick people, the loop ends. Prints out the last line and how many total people got sick and how long it took. If the infection chance is very low, there is a high chance that many people survive

```
How many people
25
Infection chance ?
.4
In step    1 #sick:   1:??????+???????????????????
In step    2 #sick:   3:?????+++???????????????????
In step    3 #sick:   5:????+++++???????????????????
In step    4 #sick:   7:???+++++++???????????????????
In step    5 #sick:   7:???+++++++???????????????????
In step    6 #sick:   6:???+++-+++???????????????????
In step    7 #sick:   4:???++---++???????????????????
In step    8 #sick:   2:???+-----+???????????????????
In step    9 #sick:   0:???-------???????????????????
Disease ran its course by step 9
7 got sick
```

```
#include <iostream>
#include <vector>
#include "person.h"
#include "population.h"
#include <ctime>
#include <cstdlib>
#include <iomanip>
using namespace std;
int main(){
int npeople;
float probs,innoc;
// Initialze  and set the values needed
cout<<"How many people ?"<<endl;
cin>>npeople;
cout<<"Infection chance ?"<<endl;
cin>>probs;
cout<<"Vaccination rate ?"<<endl;
cin>>innoc;
// first we create the population, then
innoculate
// innoculate first to prevent the innoculation
from "saving" the target
Population population(npeople,probs);
population.innoculation(innoc);
population.random_infection();
int step=0;
srand(time(NULL));
while(population.count_infected()!=0)
{
      step+=1;
      cout<< "In step "<<setw(3)<<step<<" ";
      population.Printer();
      float badluck=(float) rand() / (float)
RAND_MAX;
      population.update();
      population.setprob(badluck);
      };
cout<<"In step "<<setw(3)<<step+1<<" ";
population.Printer();
cout<<"Disease ran its course by step
"<<step+1<<endl;
cout<< population.sickcount()<<" got
sick."<<endl;
return 0;
}
```

```
How many people ?
50
Infection chance ?
.6
Vaccination rate ?
.4
In step   1 #sick:   1:??**??????*?*???***?*??????*?????**??**???*?+?*?*?
In step   2 #sick:   3:??**??????*?*???***?*??????*?????**??**???*+++*?*?
In step   3 #sick:   3:??**??????*?*???***?*??????*?????**??**???*+++*?*?
In step   4 #sick:   3:??**??????*?*???***?*??????*?????**??**???*+++*?*?
In step   5 #sick:   3:??**??????*?*???***?*??????*?????**??**???*+++*?*?
In step   6 #sick:   2:??**??????*?*???***?*??????*?????**??**???*+-+*?*?
In step   7 #sick:   0:??**??????*?*???***?*??????*?????**??**???*---*?*?
Disease ran its course by step 7
3 got sick.
```

Innoculation(innoc)

Inoculates the population is a certain percentage that is given in.

Main3.cc

First, we read in the three values. Essentially it is the same as the last program other than we inoculate a percentage of the population. Inoculated members cannot get sick anymore. Then we infect patient zero who in turn may infects adjacent members. It runs until we have no sick members.

This is an extremely inaccurate model as it assumes that the infected can only get the person next to them sick. It also allows the population to have a hard herd immunity line. If the infected is in between two inoculated they will be unable to infect anyone.

```
#include <iostream>
#include <vector>
#include "person.h"
#include "population.h"
#include <ctime>
#include <cstdlib>
using namespace std;
int main(){
int npeople;
float probs,innoc;
cout<<"How many people ?"<<endl;
cin>>npeople;
cout<<"Infection chance ?"<<endl;
cin>>probs;
cout<<"Vaccination rate ?"<<endl;
cin>>innoc;
Population population(npeople,probs);
population.innoculation(innoc);
population.random_infection();
int step=0;
srand(time(NULL));
while(population.count_infected()!=0)
{
      step+=1;
      cout<< "In step   "<<step<<" ";
      population.Printer();
      population.update();
      population.update_spread();
      };
cout<<"In step   "<<step+1<<" ";
population.Printer();
cout<<"Disease ran its course by step "<<step+1<<endl;
cout<<population.sickcount()<< " got sick"<<endl;
return 0;
}
```

Update_spread()

Spread the disease the up to 6 random people.

Main4.cc

Initializes each variable and creates a population that accepts these variables to build a population made of “Person”. We inoculate the population and randomly infect a random person in the population. It updates, then the sick person begins to have “interactions” with the rest of the population.

Generally, the population I has a distinct curve that is significantly impacted by the inoculation rate. While the infection rate is low but has a low inoculation rate as well, can infect a large amount of people still. Conversely, if the infection rate is very high, but high inoculation rate, not very many people are infected.

How many people ?

100

Infection chance ?

.5

Vaccination rate ?

.3

In step 1 #sick: 1:**???????**????????*?*??*?*??+?*?????*???****??*???*?*?*?????*??*????*????????*??????*??****??*???????????

In step 2 #sick: 4:**???????**????????*?*??*?*??+?*?+???*???****??*???*?*?+?????*??*????*????????*??????*??****??*???+???????

In step 3 #sick: 11:**????+**????????*?*??*+*??+?*?+???*??????*??*+?*?*?*+?????*?+*?+??*???+????*??????*??****??*???+??+????

In step 4 #sick: 24:**++??+**??+?????*?*??*+*??+?*?++??*?+?***??*++*?*?*+++?+*?+*?+??*???+?+???*?+?++?*??****??*???+??+????

In step 5 #sick: 44:**++??+**+++???+*?*?+*+*??++*++++?*++?**+?*++*+*?*+++++*?+*?+++?*???+?+???*?+?+++*++****?+*++?++++????

In step 6 #sick: 58:**++??+**+++?+++*?*++*+*++-+*+++++*+++**+?*++*+*?*+++++*?+*+++?*?+?+?+++*?+++++*++****++*+++++++??+?

In step 7 #sick: 60:**++??+**+++++++*+*++*+*++-+*+-+++*+++**+?*++*+*?*-++++*?+*+++?*?+++?+++*?+++++*++****++*+++-+++?+++

In step 8 #sick: 53:**++??-**+++++++*+*++*-*++-+*+-+++*+++**+?*-+*+*?*-++++*?-*+-+?*?++-?+++*?+++++*++****++*+++-++-?+++

In step 9 #sick: 40:**--??-**++-++++*+*++*-*++-+*+--++*+-+**+?*--*+*?*---+-*?-*+-+?*?++-?-++*?-+--*++****++*+++-++-?+++

In step 10 #sick: 20:**--??-**---+++-*+*+-*-*++--*----+*--+**-?*--*-*?*-----*?-*+--?*?++-?-++*?-+---*--****+-*--+----?+++

In step 11 #sick: 5:**--??-**---+---*+*--*-*----*-----*---**-?*--*-*?*-----*?-*---?*?-+-?---*?-----*--****--*-------?+-+

In step 12 #sick: 0:**--??-**-------*-*--*-*----*-----*---**-?*--*-*?*-----*?-*---?*?---?---*?-----*--****--*-------?---

Disease ran its course by step 12

64 got sick