CMSC 202 Fall 2019

Project 1 – Chase the Rabbit

**Assignment:** Project 1 – Chase the Rabbit

**Due Date:** Thursday, September 26th at 8:59pm

**Value:** 80 points

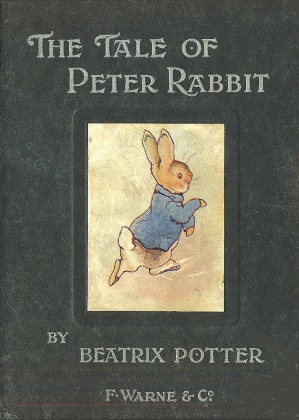
# Overview

In this project, you will:

* Practice basic C++ syntax including branching structures
* Write a program that calls multiple functions
* Manage a two-dimensional array
* Use simple file input/output

# Background

In 1902, Beatrix Potter released her first work entitled, “The Tale of Peter Rabbit” which was the first of her 23 tales. It is estimated that that book has sold 45 million copies, which puts in the top 50 best selling books of all time.



**Figure 1. The Tale of Peter Rabbit**

“The Tale of Peter Rabbit” is a children’s book that describes the story of how Peter tries to sneak into Mr. McGregor’s vegetable garden. For this project, we are going to design and implement a simple, text-based version of this story.

The vegetable garden will be represented by a two-dimensional grid. The actual implementation of how you want to create the garden is up to your design decisions, but you must meet all requirements listed below for full points.

The farmer can start anywhere in the garden searching for the rabbit. The rabbit will start somewhere randomly. While we can make the grid of any size, we will start with a standard 10 by 10.



**Figure 2. Sample Grid**

# Assignment Description

Your assignment is to develop a simple game where a farmer chases a rabbit. On each turn, the rabbit can either move in a random direction or stay where they are.

The game continues until the farmer catches the rabbit. It should then prompt the user for another game.

# Requirements:

This is a list of the requirements of this application. For this project, it is up to you exactly how you want to implement it. For you to earn all the points, however, you will need to meet all the defined requirements.

* You must follow the coding standard as defined in the CMSC 202 coding standards (found on Blackboard under course materials). This includes comments as required.
* The project must be turned in on time by the deadline listed above.
* The project must be completed in C++. You may not use any libraries or data structures that we have not learned in class. These are the only libraries that you are allowed to use in this project **<iostream>, <ctime>, <fstream>, <iomanip>, <cmath>, and <cstlib>.** You may NOT use **<string>**. You should only use **namespace std**.
* You must use a variety of functions (at least 5) including passing parameters to those functions and returning information from those functions. At least one time, an array must be passed to a function (although you may do this more than once).
* All user input must be validated. For example, if a menu allows for 1, 2, or 3 to be entered and the user enters a 4, it will re-prompt the user. However, the user is expected to always enter the correct data type. i.e. If the user is asked to enter an integer, they will. If they are asked to enter a character, they will. You do not need to worry about checking for correct data types.
* You must use at least one multi-dimensional array in this project.
* Ask the user where in the grid the farmer starts.
* Have a menu that asks if the user wants to move the farmer (north, east, south, or west)
* A way to display the farmer’s field using an R as the rabbit and the F as the farmer.
* The rabbit must be able to move every turn (but may not actually move). The rabbit may move exactly one space in any direction (north, east, south, or west) that is available. A space is available if it is inside of the limits of the grid.
* After the farmer searches in a specific direction, it indicates which direction the rabbit is (north, east, south, or west). If the rabbit is 3 west and 4 north of the farmer, it should say the rabbit is north of the farmer. If it is 3 west and 3 north, then it can say either.
* You should be able to play this with the game grid being hidden just using the directions above.
* When the farmer finds the rabbit, the game should indicate that the rabbit was found and end the game.
* Neither the farmer or the rabbit can leave the field. If the user tries to make the farmer leave the field, the user is notified that is not an option. The user is not notified if the rabbit randomly tries to leave the field and the rabbit instead does not move for that turn.
* Exit and include a thank you message for the user.
* Specific coding requirements include:

Must use at least 5 different functions.

Must use at least two different arrays.

Must pass an array to a function.

Must use at least one multidimensional array

Must pass at least one array to a function

Must not use any global variables

Must use at least one switch statement.

Must use input validation (assume the data is the correct type).

Must use at least one do..while loop.

Must use constants as needed.

# Recommendations

You are free to implement this with your own functions. While not required, these are some functions that you may want to include:

* + Start the Rabbit – randomly decides where the rabbit can start
  + Start the Farmer – user decides where to start the farmer
  + Move the Rabbit – the rabbit randomly moves one space or not
  + Move the Farmer – the farmer moves in a direction specified by the user
  + Display the Direction – used to display which direction the rabbit is in relation to the farmer
  + Print the field – used to display the entire field

# Sample Input and Output

For this project, there are no input files so every run should start the same.

In the sample output below, user input is colored blue for clarity. After compiling and running proj1, the output would look like this:

|  |
| --- |
| Welcome to Chase the Rabbit  Where would you like to start the farmer?  Enter the column(x) (0 - 9)  5  Enter the row(y) (0 - 9)  5  What would you like to do?  1. Search north.  2. Search east.  3. Search south.  4. Search west. |

Here is a long run where the rabbit runs pretty well!

|  |
| --- |
| Welcome to Chase the Rabbit  Where would you like to start the farmer?  Enter the column(x) (0 - 9)  0  Enter the row(y) (0 - 9)  0  What would you like to do?  1. Search north.  2. Search east.  3. Search south.  4. Search west.  3  The farmer moves south  The rabbit moves to another location in search of food.  The rabbit is east of the farmer.  | | | | | | | | | | |  |F| | | | | | | | | |  | | | | |R| | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  What would you like to do?  1. Search north.  2. Search east.  3. Search south.  4. Search west.  2  The farmer moves east  The rabbit moves to another location in search of food.  The rabbit is east of the farmer.  | | | | | | | | | | |  | |F| | |R| | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  What would you like to do?  1. Search north.  2. Search east.  3. Search south.  4. Search west.  2  The farmer moves east  The rabbit moves to another location in search of food.  The rabbit is east of the farmer.  | | | | |R| | | | | |  | | |F| | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  What would you like to do?  1. Search north.  2. Search east.  3. Search south.  4. Search west.  2  The farmer moves east  The rabbit nibbles on some food and does not move.  The rabbit is east of the farmer.  | | | | |R| | | | | |  | | | |F| | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  What would you like to do?  1. Search north.  2. Search east.  3. Search south.  4. Search west.  1  The farmer moves north  The rabbit nibbles on some food and does not move.  The rabbit is east of the farmer.  | | | |F|R| | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  What would you like to do?  1. Search north.  2. Search east.  3. Search south.  4. Search west.  2  The farmer moves east  The rabbit moves to another location in search of food.  The rabbit is south of the farmer.  | | | | |F| | | | | |  | | | | |R| | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  What would you like to do?  1. Search north.  2. Search east.  3. Search south.  4. Search west.  3  The farmer moves south  The rabbit moves to another location in search of food.  The rabbit is south of the farmer.  | | | | | | | | | | |  | | | | |F| | | | | |  | | | | |R| | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  What would you like to do?  1. Search north.  2. Search east.  3. Search south.  4. Search west.  3  The farmer moves south  The rabbit moves to another location in search of food.  The rabbit is south of the farmer.  | | | | | | | | | | |  | | | | | | | | | | |  | | | | |F| | | | | |  | | | | |R| | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  What would you like to do?  1. Search north.  2. Search east.  3. Search south.  4. Search west.  3  The farmer moves south  The rabbit moves to another location in search of food.  The rabbit is south of the farmer.  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | |F| | | | | |  | | | | |R| | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  What would you like to do?  1. Search north.  2. Search east.  3. Search south.  4. Search west.  3  The farmer moves south  The rabbit nibbles on some food and does not move.  The rabbit is east of the farmer.  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | |F| | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  The farmer has found the rabbit!  Congrats you've won!  Play again? (y/n) |

Here is an example validating the direction for the farmer.

|  |
| --- |
| Welcome to Chase the Rabbit  Where would you like to start the farmer?  Enter the column(x) (0 - 9)  5  Enter the row(y) (0 - 9)  5  What would you like to do?  1. Search north.  2. Search east.  3. Search south.  4. Search west.  0  1. Search north.  2. Search east.  3. Search south.  4. Search west.  5  1. Search north.  2. Search east.  3. Search south.  4. Search west. |

Here is an example where the farmer tries to leave the field (but can’t). The rabbit can’t leave the field either.

|  |
| --- |
| Where would you like to start the farmer?  Enter the column(x) (0 - 9)  0  Enter the row(y) (0 - 9)  0  What would you like to do?  1. Search north.  2. Search east.  3. Search south.  4. Search west.  1  The farmer can't leave the field  The rabbit moves to another location in search of food.  The rabbit is south of the farmer.  |F| | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  |R| | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  What would you like to do?  1. Search north.  2. Search east.  3. Search south.  4. Search west.  4  The farmer can't leave the field  The rabbit moves to another location in search of food.  The rabbit is south of the farmer.  |F| | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  | | | | | | | | | | |  |R| | | | | | | | | |  | | | | | | | | | | |  What would you like to do?  1. Search north.  2. Search east.  3. Search south.  4. Search west. |

# Compiling and Running

To compile your program, enter the following command at the Linux prompt:

g++ -Wall proj1.cpp -o proj1 (use this command to show warnings – which should be eliminated before turning your code in)

This command runs the GNU C++ compiler (**g++**). The option **-Wall** instructs the compiler to be verbose in its production of warning messages; the option **-o proj1** (hyphen followed by the letter "o", not the digit zero), instructs the compiler to give the executable program the name **proj1**. If the program compiles correctly, the executable file **proj1** will be created in the current directory. Your project files should have no warnings or errors when turned in.

At the Linux prompt, enter the command **./proj1** to run your program. It should look like the sample output provided above.

# Completing your Project

When you have completed your project, you can copy it into the submission folder. You can copy your files into the submission folder as many times as you like (before the due date). We will only grade what is in your submission folder.

For this project, you should submit these files to the **proj1** subdirectory:

**proj1.cpp** — should include your implementations of the required functions.

Submitting your project has two steps:

1. Set up a symbolic link in your home directory to your submission folder. Execute these commands:
   1. **cd ~**
   2. For the next command, copy it exactly – you should not need to modify it at all (**$USER** will automatically populate your user name on GL). Also, notice the space after **$USER** and before **~/cs202proj**.

**ln -s /afs/umbc.edu/users/j/d/jdixon/pub/cmsc202/$USER ~/cs202proj**

* 1. To check that the symbolic link was built successfully, you can type:
     1. **ls ~/cs202proj**
     2. **It should list proj1, proj1-late1, proj1-late2 through proj5-late2**

1. Copy the project files into your proj1 folder. Execute these commands:
   1. cd to your project 1 folder. An example might be:

**cd ~/202/projects/proj1**

* 1. **cp proj1.cpp ~/cs202proj/proj1**

You can check to make sure that your files were successfully copied over to the submission directory by entering the command

ls ~/cs202proj/proj1

You can check that your program compiles and runs in the **proj1** directory, but please clean up any **.o** and executable files. Again, do not develop your code in this directory and you should not have the only copy of your program here.

**IMPORTANT:** If you want to submit the project late (after the due date), you will need to copy your files to the appropriate late folder. If you can no longer copy the files into the proj1 folder, it is because the due date has passed. You should be able to see your proj1 files but you can no longer edit or copy the files in to your proj1 folder. (They will be read only)

* If it is 0-24 hours late, copy your files to **~/cs202proj/proj1-late1**
* If it is 24-48 hours late, copy your files to **~/cs202proj/proj1-late2**
* If it is after 48 hours late, it is too late to be submitted.