Adventures in Programming

CS151 – Programming Assignment 1

Fall 2017

# Introduction

Colossal Cave Adventure was the first interactive fiction game ever written (circa 1975). It was originally created by a programmer by the name of Will Crowther, who had spent time exploring Mammoth Cave in Kentucky. Don Woods, a graduate student at Stanford University made a large number of improvements to the game in 1976. Woods frequently gave the game as a project for his first year computer science students at Stanford University[[1]](#footnote-1).

Since the ancient days of Colossal Cave, programmers and game companies have developed many different adventure and interactive fiction games. Some of the more famous of these games were ZORK, Myst and King’s Quest. For this first part of the assignment, you will be creating an adventure type game that has a main character. In subsequent assignments throughout this course, we will expand the game to have some form of conflict so that it will take on more of a role-playing feel[[2]](#footnote-2).

# Game Play

Adventure was an entirely text based game. The program would give a description of the user’s surroundings and provide the user with an input prompt. The user could then enter a words or a phrase to interact with the world. For example, the user could enter the command “NE” to have the character move NE. The game would then describe the location of the character.

The user could move throughout the world (N, S, E, W, NE, NW, SE, SW, IN, OUT, UP, DOWN), interact with different items (GET, DROP, USE, etc.) and sometimes with characters (such as a pirate, a bird or a dwarf). Having certain items in your inventory would help you to accomplish specific tasks. For example, you could use the cage to capture the bird or kill the dwarves with an axe. If Adventure did not understand a keyword, it would reply that it did not understand.

As they moved through the world, the user would score points based on their actions up to a maximum of 350. The user could also quit at any time to receive their final score.

# Program Requirements

For the purposes of this course, you will need to make your own simplified version of Adventure, which will include several of the elements from the original game. In particular, your game will need to meet the following minimum requirements.

* An overall objective. For example, get the character to Room101.
* A point based system. For example, you can give points for visiting locations or for interacting with certain parts of the world.
* A character. At a minimum, the character should be a class that has member variables for their name and game score. Of course, you can add more member variables if you want to make your game more complicated.
* A map of your world that shows the dependencies between the locations in the world. From a design perspective, you should have a location class with a description for each location. You will need to have at least 25 locations in your world. You need to implement the entire world as a two dimensional array and allow the user to navigate through it.
* At least three locations should support some form of interaction. For example, you could have a fountain from which the character can drink or a simple puzzle they need to solve to finish the game. This can be implemented as a function.
* A list of commands that the program will accept. You are free to implement as many commands as you like. At a minimum, you will need the following commands:
  + Help – Display help for the user
  + N,S,E,W – The character should be able to move in one of four directions. This should change their location unless they are on the edge of the map or move to an invalid location.
  + Score – The program should display the score.
* Constructors, accessor, and mutator functions for all your classes. You may also want some friend functions to help you with input and output.
* Separate Compilation – Your program should support separate compilation with header files for each class.

# Part 1 – Understand the Problem (10 Points)

There are several versions of adventure available on the Internet. You can play online at: <http://www.amc.com/shows/halt-and-catch-fire/exclusives/colossal-cave-adventure>. Play this version of Adventure and report your score on the Laulima message board dedicated to the Adventure Game before **August 30, 2017.**

# Part 2 – Create a Plan (20 Points)

After understanding the problem, you will create a plan for your program. Since this program is much more complicated than the ones you created in CS150, it is necessary to have a plan. To make sure that you can implement the program; your plan should include the following items:

* + A clearly stated objective for the program.
  + The **variables** that your program will use and how you will name and use them. These variables should include the classes that your program needs to have if you are using an object oriented approach.
  + The **major functions** your program will have along with descriptions of their parameters. You may want to identify pre-conditions and post-conditions.
  + Descriptions of the **algorithms** your program will use.

Don’t be afraid to use visual representations and diagrams in your plan. For example, you may want to include a map of your adventure world[[3]](#footnote-3).

I will select one or two students to present their plans in front of the class. To make your plan easier to present, you should use Microsoft PowerPoint to organize your ideas.

Have your plan ready by **August 30, 2017**.

# Part 3 – Implement Your Program (60 Points)

For this part of the assignment, you will implement your plan in code using C++. Your code should have appropriate comments to explain the program and the code. It should be written in a consistent and readable form and compile without errors or warnings.

**Warning**: This game requires far more code than anything you created in CS150. **Start Early!**

**Submit your code by September 11, 2017.**

# Part 4 – Testing (10 Points)

Create a simple functional test plan. This can be a walkthrough[[4]](#footnote-4) that shows the steps to solve your game. Submit your test plan along with your code using the assignment link on **September 11, 2017**.

# Extra Credit and Bonus (up to 15 Points)

Post an executable version of your game to the Laulima adventure forum as its own topic, e.g. Keith’s Adventure Game. You will need to zip the file in order to have Laulima accept it as an attachment on the message boards. Do not submit your source code, just the executable.

Test at least three other student’s game and give it a rating from 1-10 as a response to the original topic before **September 18, 2017**.

The two highest rated games based on student ratings and instructor evaluation will receive a bonus to their grade.

1st Place – 15 Bonus Points + Prize  
2nd Place – 10 Bonus Points

All testers receive 3 bonus points to their grade.

1. You can read more about this game at: <http://www.rickadams.org/adventure/index.html> [↑](#footnote-ref-1)
2. Of course, the conflict can be non-violent and you can be very creative in your conflict resolution strategies. [↑](#footnote-ref-2)
3. See <http://www.rickadams.org/adventure/maps/advent1.gif> as an example. Your program will not be as complicated as many of the example maps. [↑](#footnote-ref-3)
4. See <http://www.rickadams.org/adventure/walkthroughs/walkthrough.html> as an example. Obviously, your program will not be this complicated. [↑](#footnote-ref-4)