Choose Your Own Adventure

CS 002 Honors Project

Project Requirements Specification

Version Number 3

Prepared by:

Derrik Luong
Kevin Truong
Zihe Liu

Date

05/06/2024 - 06/10/2024

Changelog Template

1. 05/06/2024
    a. Discussed potential user experience improvements based on playtesting.
    b. Planned enhancements to make the game more engaging.
2. 05/13/2024
    a. Identified areas for improvement and optimization.
3. 05/20/2024
    a. Improvement of all branches
    b. Discuss presentation
4. 05/27/2024
    a. Updated project documentation to reflect new features and changes.
    b. Added code for character battles, including health bars and attack mechanics.
5. 06/04/2024
    a. Identified and fixed bugs related to path navigation and event triggers.
    b. Ensured all paths are fully functional and lead to valid end states.

Weekly Status Report[1]

05/06/2024-05/12/2024

Project Name: Choose Your Own Adventure

Name(s) of Participants: [Derrik, Kevin, Zihe]

Participant(s) Role(s): [Derrik] - [Analyst]
[Kevin] - [Developer]
[Zihe] - [Researcher]

1. Status of actionable items from the previous weekly report
    a. Actionable item description and status:
        i. Checking the operation of the project
        ii. 2hrs
        iii. everyone

2. Actionable items for the current week
    a. Actionable item description and status:
        i. Developing a plan for the final phase
        ii. 05/12/2024
        iii. everyone
3. Issues
    a. N/A
4. Additional Notes

[1] One should be completed for every week, chronicled here.

Weekly Status Report[2]

05/13/2024-05/19/2024

Project Name:	Choose Your Own Adventure
Name(s) of Participants:	[Derrik, Kevin, Zihe]
Participant(s) Role(s):	[Derrik] - [Analyst] [Kevin] - [Developer] [Zihe] - [Researcher]

5. Status of actionable items from the previous weekly report
    a. Actionable item description and status:
        i. Refinement of shortcomings in the project
        ii. 3hrs
        iii. everyone

6. Actionable items for the current week
    a. Actionable item description and status:
        i. Continue to refine the design of each route
        ii. 05/19/2024
        iii. everyone
7. Issues
    a. N/A
8. Additional Notes

[2] One should be completed for every week, chronicled here.

Weekly Status Report[3]

05/20/2024-05/26/2024

Project Name:	Choose Your Own Adventure
Name(s) of Participants:	[Derrik, Kevin, Zihe]
Participant(s) Role(s):	[Derrik] - [Analyst] [Kevin] - [Developer] [Zihe] - [Researcher]

9. Status of actionable items from the previous weekly report
    a. Actionable item description and status:
        i. Talk about the user's post-use experience and make plans for improving the program
        ii. 3hrs
        iii. everyone

10. Actionable items for the current week
    a. Actionable item description and status:
        i. Improvement of all branches
        ii. 06/10/2024
        iii. Everyone
    b. Actionable item description and status:
        i. Discuss presentation time slots
        ii. 5 minutes
        iii. Everyone
11. Issues
    a. N/A
12. Additional Notes

[3] One should be completed for every week, chronicled here.

Weekly Status Report[4]

05/27/2024-06/03/2024

Project Name:	Choose Your Own Adventure
Name(s) of Participants:	[Derrik, Kevin, Zihe]
Participant(s) Role(s):	[Derrik] - [Analyst] [Kevin] - [Developer] [Zihe] - [Researcher]

13. Status of actionable items from the previous weekly report
    a. Actionable item description and status:
        i. Finish all paths of the adventure
        ii. 4hrs
        iii. Zihe Liu
    b. Actionable item description and status:
        i. Almost done with the battle system
        ii. 4hrs
        iii. Kevin
14. Actionable items for the current week
    a. Actionable item description and status:
        i. Talk about when we are going to finish all our work and finish our project basically.
        ii. 06/03/2024
        iii. everyone
15. Issues
    a. N/A
16. Additional Notes

[4] One should be completed for every week, chronicled here.

Weekly Status Report[5]

06/04/2024-06/10/2024

Project Name:	Choose Your Own Adventure
Name(s) of Participants:	[Derrik, Kevin, Zihe]
Participant(s) Role(s):	[Derrik] - [Analyst] [Kevin] - [Developer] [Zihe] - [Researcher]

17. Status of actionable items from the previous weekly report
    a. Actionable item description and status:
        i. Work on the presentation
        ii. 5hrs
        iii. everyone
    b. Actionable item description and status:
        i. Finish the battle system
        ii. 3hrs
        iii. Kevin
    c. Actionable item description and status:
        i. Add battle events for bandits, bats, and boa and program testing
        ii. 1hr
        iii. Derrik

18. Actionable items for the current week
    a. Actionable item description and status:
        i. Finish the project and the presentation
        ii. 06/10/2024
        iii. Everyone
19. Issues
    a. N/A
20. Additional Notes

[5] One should be completed for every week, chronicled here.

Name	Date	Reason for Changes	Version
Kevin Truong	3/27/2024	Made a system that will allow for easy implementation of different branching paths in future versions	1.0.0
Kevin Truong	4/29/2024	Started creating the “North” path	1.1.0
Zihe Liu	5/1/2024	Further refinement of the code	1.1.1
Zihe Liu	5/5/2024	More development on the West and South paths	2.0.0
Zihe Liu	5/14/2024	North Path is finished	2.0.1
Kevin Truong	5/20/2024	Battle system is implemented	2.1.0
Zihe Liu	06/02/2024	Completion of all branches	3.0.0
Kevin Truong	06/04/2024	Battles for certain paths are added	3.1.0
Derrik Luong	06/10/2024	Added more battles	3.1.1
Kevin Truong	06/11/2024	Game is polished, bugs are fixed	3.1.2

Introduction

Purpose

The project is an adventure story game created with C++ programming. This program requires the user to select choices and battle NPCs in order to escape the dungeon (the main objective of the story). The purpose of this game is to expand to the genre of entertainment pertaining to offline story games. This game allows players to immerse themselves into a story through a simple program.

Document Conventions

Programming Language: C++
IDE: Visual Studio
Operating System: Windows 10
Game Language: English
Dependencies: Standard C++ libraries, Windows-specific headers

Intended Audience and Reading Suggestion

This documentation is intended for developers, project managers, and users interested in understanding the development and features of the adventure story game. It benefits team members involved in the project, potential contributors, and anyone interested in the technical and design aspects of the game.

Project Scope

The project encompasses the development of an interactive adventure story game where players navigate through a dungeon by making choices and engaging in battles with NPCs.

References

Problem Solving with C++: Chapter 7
https://www.geeksforgeeks.org/sleep-function-in-cpp/
https://www.linkedin.com/pulse/can-objects-member-function-modify-another-object-c-konstantin-rebrov/

Overall Description

Project Features

The game offers multiple branching paths, each leading to unique scenarios and endings. Players must make decisions that influence the direction and outcome of their adventure. Players engage in battles with various NPCs (Non-Player Characters). The battle system includes health points, damage, and accuracy mechanics.The game presents an engaging narrative that guides players through a mysterious dungeon. The story unfolds based on the choices players make.

User Characteristics

Individuals who enjoy offline, story-driven games with interactive elements.Engage with the game by making choices, battling NPCs, and exploring different paths.

Operating Environment

Users can run the game on Windows 10 systems. The game does not require internet access, making it ideal for offline play.

Design and Implementation Constraints

- What computer(s) did you use?
  Windows 10
- What language(s) did you use?
  We used a C++ as the primary programming language.
  As of now, the game only has English as the main language.
- What IDE did you use?
  Visual Studio was used.

Assumptions and Dependencies

What are the system and user requirements for someone to get the most out of your project?

- Do they need internet? No
- Do they need basic reading skills? Yes

System Features[6]

Use Case (Number): Battle System	
Objective:	This feature allows users to battle enemies, requiring them to defeat the enemies to continue.
Priority:	High
Flow of Events: The user carries out this feature when encountering certain routes.	

Use Case (Number): Multiple Choice Navigation	
Objective:	This feature allows users to make choices that determine the direction and outcome of their adventure.
Priority:	High
Flow of Events: The player is presented with a scenario and multiple choices and selects an option using the multiple-choice prompt.	

Use Case (Number): Health Bar Display	
Objective:	This feature visually represents the health status of the player and enemies during battles.
Priority:	Medium
The health bar is displayed at the beginning of a battle. As the player or enemy takes damage, the health bar updates to reflect the current health status. The battle continues until one of the health bars is depleted.	

[6] You will be making ‘cards’ based on the template provided for each of the use cases

Code

```
////////////////////////////////////////////////////////////////////////////
//////
// Title:            Group 1 Iteration 3
// Files:            group1_iteration3.cpp
// Semester:         Spring 2023
//
// Members:          Kevin, Zihe, Derrik
// Lecturer's Name:  Hilario Balajadia
// Version:          1.1.0
// Brief:            This program allows the user to play through a
                    //  choose-your-own adventure styled game.
////////////////////////////////////////////////////////////////////////////
//////


////////////////////////////////////////////////////////////////////////////
//////
// Online sources
//     Problem Solving with C++: Chapter 7
//     https://www.geeksforgeeks.org/sleep-function-in-cpp/
//
https://www.linkedin.com/pulse/can-objects-member-function-modify-another-
object-c-konstantin-rebrov/
////////////////////////////////////////////////////////////////////////////
//////



#include <iostream>
```

```
#include <string>
#include <windows.h>
#include <cstdlib>

// global array
const char multipleChoiceLetters[5] = {'A','B','C','D','E'};

class Character
{
    public:
        std::string name;
        int hitpoints, maxHitpoints, damage, accuracy;

    // Object constructor
    Character(std::string theName, int theMaxHitpoints, int theDamage, int
theAccuracy)
    {
        name = theName;
        hitpoints = theMaxHitpoints;
        maxHitpoints = theMaxHitpoints;
        damage = theDamage;
        accuracy = theAccuracy;
    }

    // Used when a character successfully attacks another character
    void attack(Character* other)
    {
        other->hitpoints -= damage;
    }

    // Prints the healthbar of the character
    void printHealthbar()
    {
        std::cout << "|";
        for (int i = 1; i <= maxHitpoints; i++)
        {
            if (i <= hitpoints)
            {
                std::cout << "=";
            }
```

```
            else
            {
                std::cout << " ";
            }
        }
        std::cout << "|";
    }
};

void gameOver();
void youWin();
void error();
void waitFor(double seconds);
char multipleChoicePrompt(std::string choiceA, std::string choiceB);
char multipleChoicePrompt(std::string choiceA, std::string choiceB,
std::string choiceC);
char multipleChoicePrompt(std::string choiceA, std::string choiceB,
std::string choiceC, std::string choiceD);
char multipleChoicePrompt(std::string choiceA, std::string choiceB,
std::string choiceC, std::string choiceD, std::string choiceE);
int randomNumberBetween(int firstNumber, int secondNumber);
void Battle(Character& character1, Character& character2);
void displayHealthbars(Character& character1, Character& character2);
void characterAttack(Character& attacker, Character& defender);
void characterHeal(Character& theCharacter);

int main()
{
    srand(time(NULL));
    Character player("Player", 20, 8, 80);

    // Prologue
    std::cout << "You wake up one day, lost, confused, and stranded in a
mysterious and dark cave.";
    waitFor(5);
    std::cout << "\n";
    std::cout << "You don't know how you got there, but all you can think
about is seeing the sunlight again.";
    waitFor(5);
    std::cout << "\n";
```

```
    std::cout << "You call out, but to no avail, there is no response.";
    waitFor(4);
    std::cout << "\n";

    std::cout << "There are four chambers going in four directions, as
well as a strange staircase leading down.";
    waitFor(5);
    std::cout << "\n\n";
    std::cout << "Which way do you head?\n";
    char choice;
    choice = multipleChoicePrompt("North", "South", "East", "West",
"Stairs");
    std::cout << "\n";
    switch (choice)
    {
        case 'A': // User chooses "North"
        {
            std::cout << "As you head North, the dungeon seems to lead you
outside. you are suddenly met with a fierce and cold snow storm. \n";
            std::cout << "You see a nearby cave, and consider whether you
should keep moving or wait the storm out. \n";
            choice = multipleChoicePrompt("Continue walking in the storm",
"Take shelter in the nearby cave");
            std::cout << "\n";
            switch (choice)
            {
                case 'A': // User chooses "Continue walking in the storm"
                {
                    std::cout << "You decide to tough it out and walk
through the storm. You walk for hours, but there appears to be nothing but
snow for miles. \n";
                    std::cout << "You eventually collapse from exhaustion,
freezing to death. \n";
                    gameOver();
                }
                case 'B': // User chooses "Take shelter in the nearby
cave"
                {
                    std::cout << "The storm passes and you are able to
safely continue. ";
```

```
                    std::cout << "However, you spot a pair of vicious
looking penguins. ";
                    std::cout << "How do you want to proceed?\n";
                    choice = multipleChoicePrompt("Attempt a sneak
attack", "Try to go around them", "Wait for them to leave");
                    std::cout << "\n";
                    switch (choice)
                    {
                        case 'A': // User chooses "Attempt a sneak attack"
                        {
                            int sneakAttackAttempt =
randomNumberBetween(1, 100);
                            if (sneakAttackAttempt >= 80)
                            {
                                std::cout << "The sneak attack was
successful! The penguins become frightened, and they frantically run off
into the distance. ";
                                std::cout << "You continue trekking, and
eventually make it back to civilization.\n";
                                youWin();
                            }
                            else
                            {
                                std::cout << "The sneak attack failed, and
now the penguins are super angry!\n";
                                waitFor(3);
                                for (int i = 0; i < 2; i++)
                                {
                                    Character penguin("Angry Penguin", 15,
9, 55);

                                    Battle(player, penguin);
                                }
                                std::cout << "With the penguins defeated,
you quickly make your escape. \n";
                                std::cout << "Have long hours of
traveling, you eventually make it back to civilization.";
                                youWin();
                            }
                        }
                        case 'B': // User chooses "Try to go around them"
```

```
                        {
                            std::cout << "Just before you are able to walk
by unnoticed, one of the penguins sees you and alerts the other of your
presence.\n";
                            waitFor(3);
                            for (int i = 0; i < 2; i++)
                            {
                                Character penguin("Penguin", 15, 6, 65);
                                Battle(player, penguin);
                            }
                            std::cout << "Having gained their respect, the
penguins bow down to you. \n";
                            std::cout << "One of them notices you look
lost, and pulls out a map.\n";
                            std::cout << "You follow the map, and it leads
you back to civilization.\n";
                            youWin();
                        }
                        case 'C': // User chooses "Wait for them to leave"
                        {
                            std::cout << "You decide to stay where you are
and wait until the penguins leave. \n";
                            std::cout << "One of the penguins pull out a
deck of cards, and they start playing Go Fish. \n";
                            std::cout << "As you wait, minutes turn to
hours, and hours turn to days.";
                            std::cout << "You eventually starve to death
as the penguins start another round.\n";
                            gameOver();
                        }
                        default:
                        {
                            error();
                        }
                    }
                }
                default:
                {
                    error();
                }
```

```
            }
        }
        // User chooses "South"
        case 'B':
        {
            std::cout << "You go south and spot something in the corner of
your eye. An odd rock in the wall catches your attention and you decide to
analyze the area.\n";
            std::cout << "When you touch the rock, the ground opens up and
dumps you at the entrance of a cave. It is a bit odd how this cave seems
well-laid out and put together, but you proceed anyways.\n";
            choice = multipleChoicePrompt("Take the minecart", "Climb down
the rocks"," Climb up ladder");
            std::cout << "\n";
            switch (choice)
            {
                case 'A': // User chooses "Take the minecart"
                {
                    std::cout << "You decide to explore the cave in a mine
cart, and after the cart has been running for a while, you notice two
paths appearing ahead of you. \n";
                    choice = multipleChoicePrompt(" Take left rail", "
Take right rail");
                    std::cout << "\n";
                    switch (choice)
                    {
                        case 'A':
                        {
                            std::cout << "You choose the left rail,
unfortunately, you did not see the \"CONSTRUCTION IN PROGRESS SIGN\" and
the minecart reaches the end of the rail and slams you against the cavern
wall.\n";
                            gameOver();
                        }
                        case 'B':
                        {
                            std::cout << "You choose the right rail, It
seems that the dungeon's creator used this minecart to get to and out of
the dungeon.\n";
                            std::cout << "You escape the dungeon.\n";
```

```
                        youWin();
                    }
                    default:
                    {
                        error();
                    }
                }
            }
            case 'B': // User chooses "Climb down the rocks"
            {
                std::cout << "You choose to climb down, and after some
time has passed, you feel it getting hotter and hotter, and you look down
in horror. \n";
                std::cout << "You realize that directly below you is a
large pool of lava, and the amazing heat makes you feel fearful. \n";
                std::cout << "However, your strength is exhausted, and
you are afraid that you can no longer return, and at last you fall into
the lava in an endless fit of remorse. \n";
                std::cout << "You died swimming in lava.\n";
                gameOver();
            }
            case 'C': // User chooses "Climb up ladder"
            {
                std::cout << "You choose to climb the ladder, and
after who knows how long, you come to another platform. \n";
                std::cout << "Surprisingly there is a faint light
coming from the cave ahead of you, but to reach it you need to cross a
large pool of water.\n";
                choice = multipleChoicePrompt("Dive into pool of
water", "Make a raft");
                switch (choice)
                {
                    case 'A': // User chooses "Dive into pool of
water"
                    {
                        std::cout << "You choose to dive and
unfortunately your feet get caught in the water plants and you end up
drowning.\n";
                        gameOver();
                    }
```

```
                        case 'B': // User chooses "Make a raft"
                        {
                            std::cout << "You choose to make a raft, and
luckily you find quite a few sturdy materials around. After half a day of
crafting, you arrive in front of the cave on the raft. \n";
                            std::cout << "You look into the cave, only to
realize that the faint light is being emitted by a monster covered in
scales.\n";
                            std::cout << "Your arrival woke it up. You
died, of being torn apart.\n";
                            gameOver();
                        }
                        default:
                        {
                            error();
                        }
                    }
                }
            }
        }
        case 'C': // User chooses "East"
        {
            std::cout << "You decide to head east, confident that this
path will lead you out of the dungeon. As you approach, you see the
entrance to a labyrinthine maze, its walls towering high above you and
casting long shadows.\n";
            std::cout << "You take a deep breath and step inside.\n\n";
            waitFor(3);

            // Enter the maze and face the first decision point
            std::cout << "You step into the maze, the cold stone underfoot
echoing your footsteps. The air is damp and the only light comes from your
flickering torch. You look ahead and see three paths: straight, left, and
right.\n";
            choice = multipleChoicePrompt("Continue straight ahead", "Head
left", "Head right");
            std::cout << "\n";

            switch (choice)
            {
```

```
                case 'A':  // User chooses "Continue straight ahead"
                {
                    std::cout << "You move forward, the path narrowing.
After a few steps, you reach a dead end with a strange rune etched into
the wall. It seems to hum with an unknown power. \n";
                    std::cout << "You continue pushing forward, but you
feel your legs get tired as your mind grows drowsy.\n";
                    std::cout << "You eventually decide to turn back, but
just as you turn the walls start to glow, draining your remaining energy
and causing you to collapse.\n";
                    gameOver();
                }
                case 'B':  // User chooses "Head left"
                {
                    std::cout << "You turn left, the passageway winding
and twisting. You hear the faint sound of water dripping somewhere in the
distance.\n";
                    waitFor(3);
                    std::cout << "You turn left and continue down the
corridor. The walls are covered in vines, and you brush them aside as you
walk. After a few minutes, you reach another fork in the path.\n";
                    choice = multipleChoicePrompt("Take the left path",
"Take the right path");
                    std::cout << "\n";

                    switch (choice)
                    {
                        case 'A':  // User chooses "Take the left path"
                        {
                            std::cout << "You take the left path and find
yourself in a small chamber. In the center is a pedestal with an ancient,
dusty tome. You pick it up, and inside, you find a map of the maze! With
renewed hope, you study it carefully.\n";
                            waitFor(3);

                            // Using the map to navigate
                            std::cout << "Using the map, you see that the
correct path involves taking the next left, then heading straight for a
while before turning right. You follow the map's guidance.\n";
                            waitFor(3);
```

```
                            std::cout << "You take the left turn as
indicated, walking for what seems like hours. The maze is eerily silent
except for the occasional distant rumble.\n";
                            waitFor(3);
                            std::cout << "You reach another intersection
and consult the map. You turn right, the path gradually sloping
upwards.\n";
                            waitFor(3);

                            // Encounter with the old woman
                            std::cout << "As you progress, you hear a
soft, melodic humming. You follow the sound and find a small, luminescent
pool with a figure standing beside it. It's an old woman, her eyes glowing
with an otherworldly light.\n";
                            std::cout << "\"Welcome, traveler. You have
come far. This maze was built to test the resolve of those who enter. I
can offer you guidance,\" she says.\n";
                            waitFor(3);

                            // Following the old woman's guidance
                            std::cout << "With the old woman's help, you
navigate through the last few twists and turns of the maze. You face a few
minor challenges, but nothing you can't handle.\n";

                            // Battle with boa constrictor
                            std::cout << "Suddenly, you are confronted by
a giant boa constrictor!\n";
                            waitFor(3);
                            Character boa("Boa Constrictor", 13, 7, 85);
                            Battle(player, boa);
                            std::cout << "You engage in a fierce battle
with the boa constrictor and manage to defeat it.\n";
                            waitFor(3);

                            // Battle with stone guard
                            std::cout << "You continue on your path, but
soon you encounter a stone guard blocking your way.\n";
                            waitFor(3);
                            Character stoneGuard("Stone Guard", 30, 10,
30);
```

```
                            Battle(player, stoneGuard);
                            std::cout << "After a tough battle, you manage
to defeat the stone guard.\n";
                            waitFor(3);

                            // Battle with blood-sucking bat
                            std::cout << "As you move forward, a swarm of
blood-sucking bats swoops down on you!\n";
                            waitFor(3);
                            for (int i = 0; i < 3; i++)
                            {
                                Character bat("Blood-sucking Bat", 4, 4,
100);
                                Battle(player, bat);
                            }
                            std::cout << "You successfully defeat the
blood-sucking bats after a vicious fight.\n";
                            waitFor(3);

                            std::cout << "With the old woman's guidance
and your battles won, you navigate through the last few twists and turns
of the maze. You reach a final door, intricately carved with symbols of
protection and escape. You push it open, the heavy stone grinding against
the floor.\n";
                            waitFor(3);
                            std::cout << "As you step through the door,
you find yourself outside the maze, the cool breeze of freedom washing
over you. You've made it out of the east maze.\n";
                            waitFor(3);
                            youWin();
                        }
                        case 'B':  // User chooses "Take the right path"
                        {
                            std::cout << "You turn right and find yourself
in a room filled with cobwebs. As you move through, you disturb a nest of
giant spiders. \n";
                            Character spider("Mother Spider", 25, 7, 60);
                            Battle(player, spider);
```

```
                            std::cout << "After defeating the giant
spider, you continue going forward, running from the Mother Spider's
furious children.\n";
                            std::cout << "You soon find a vine that you
are able to use to climb up. As you continue climbing, you look up and see
light.\n";
                            std::cout << "After a little more climbing,
you crawl through the hole in the ceiling. Back in the overworld, you head
back to civilization. \n";
                                youWin();
                            }
                            default:
                            {
                                error();
                            }
                        }
                        break;
                    }
                    case 'C':  // User chooses "Head right"
                    {
                        std::cout << "You turn right and immediately fall
through a trapdoor, where some nice, shiny spikes are ready to greet you.
\n";
                        gameOver();
                    }
                    default:
                    {
                        error();
                    }
                }
                break;
            }
            case 'D': // User chooses "West"
            {
                std::cout << "After countless days of city walk, you spot a
small underground city. For a village within a dungeon, you think to
yourself that this seems like any other ordinary place. You decide to rest
here for a while, but first you decide to...\n";
                choice = multipleChoicePrompt("Explore the village", "Turn
into a passageway");
```

```
            std::cout << "\n";
            switch (choice)
            {
                case 'A': // User chooses "Explore the village"
                {
                    std::cout << "You spot some (friendly looking) locals
and so you decide to talk with them. They tell you that they have
encountered other lost adventurers such as yourself before.\n";
                    std::cout << "One of the villagers lends you their
horse and tells you how to return back to civilization.\n";
                    youWin();
                }
                // User chooses "Turn into a passageway"
                case 'B':
                {
                    std::cout << "As you walk into the passageway, a group
of bandits surround you. \n";
                    choice = multipleChoicePrompt("Engage in combat",
"Run");
                    std::cout << "\n";
                    switch (choice)
                    {
                        case 'A':
                        {
                            waitFor(1);
                            Character bandit("Hostile Bandits", 20, 8,
65);
                            Battle(player, bandit);
                            std::cout << "Miraculously, you defeat the
bandits despite the odds!\n";
                            std::cout << "Unfortunately, you are too
injured to continue, and collapse from your injuries.\n";
                            gameOver();
                        }
                        case 'B': // User chooses "Run"
                        {
                            // 60% chance of fleeing
                            if ((rand() % 100) < 60)
                            {
```

```
                                std::cout << "You manage to escape the
bandits and live another day.\n";
                                std::cout << "A family lets you hide in
their home.\n";
                                std::cout << "You plan to only stay for a
few days, but as you continue living with them you grow fond of your new
life.\n";
                                std::cout << "You decide to settle down
and live the rest of your days in the village.\n";
                                youWin();
                            }
                            else
                            {
                                std::cout << "The bandits caught up to
you...\n";
                                gameOver();
                            }
                        }
                        default:
                        {
                            error();
                        }
                    }
                    break;
                }
                default:
                {
                    error();
                }
            }
        }
        // User chooses "Stairs"
        case 'E':  // User chooses "Stairs"
        {
            std::cout << "You notice a dark and dim spot in the corner of
the dungeon. This intrigues you, so you decide to inspect this area of the
dungeon first. You discover an odd wooden board in the ground that closes
off some kind of area. You lift it off and find a set of underground
stairs.\n";
            std::cout << "\n";
```

```
            waitFor(3);

            // Enter the stairs and face the first decision point
            std::cout << "You descend the stairs carefully, the air
growing colder and damper as you go deeper. The light from your torch
flickers against the stone walls, casting eerie shadows. At the bottom,
you find yourself at a crossroads with two options: a door marked 'Danger'
and a slide leading into darkness.\n";
            choice = multipleChoicePrompt("Open the Danger door", "Slide
down the Hole");
            std::cout << "\n";

            switch (choice)
            {
                case 'A':  // User chooses "Danger door"
                {
                    std::cout << "You gather your courage and decide to
open the door marked 'DANGER'. The door creaks open, revealing a narrow
passageway. You walk through it cautiously, and after a few tense moments,
you see light at the end of the tunnel. You push forward and find yourself
outside the dungeon, having successfully escaped through the danger
door.\n";
                    waitFor(3);
                    youWin();
                }
                case 'B':  // User chooses "Hole Slide"
                {
                    std::cout << "You decide to take a leap of faith and
slide down the hole. The slide twists and turns, speeding up as you go.
After what feels like an eternity, you land right into a pit of lava. \n";
                    waitFor(3);
                    gameOver();
                    break;
                }
                default:
                {
                    error();
                }
            }
        }
```

```
    }
    exit(1);
}


    // choice = multipleChoicePrompt("", "______", "______", "______",
"______");
    // std::cout << "\n";
    // switch (choice)
    // {
    //     case 'A':
    //     {

    //         break;
    //     }
    //     case 'B':
    //     {

    //         break;
    //     }
    //     default:
    //     {
    //         std::cout << "Sorry, that is not a valid input.";
    //         exit(1);
    //     }
    // }



// Used when the user gets defeated in battle or chooses an event that
ends the game
void gameOver()
{
    std::cout << "GAME OVER";
    exit(0);
}

// Used when the user reaches the end of a path without losing or getting
defeated in battle
void youWin()
```

```
{
    std::cout << "YOU WIN";
    exit(0);
}

// Used when the user enters an invalid input
void error()
{
    std::cout << "Invalid option. Please restart the game.\n";
    exit(1);
}

// Delays program for a set amount of seconds
void waitFor(double seconds)
{
    double secondsToMilliseconds = seconds * 1000;
    Sleep(secondsToMilliseconds);
}

// prints the multiple choice options given strings as arguments
char multipleChoicePrompt(std::string choiceA, std::string choiceB)
{
    std::string choiceArray[2] = {choiceA, choiceB};
    for (int i = 0; i <= 1; i++)
    {
        std::cout << multipleChoiceLetters[i] << ") " << choiceArray[i] <<
"\n";
    }
    std::cout << "\n";

    std::cout << "Your choice: ";
    char decision;
    std::cin >> decision;
    return decision;
}
char multipleChoicePrompt(std::string choiceA, std::string choiceB,
std::string choiceC)
{
    std::string choiceArray[3] = {choiceA, choiceB, choiceC};
    for (int i = 0; i <= 2; i++)
```

```
    {
        std::cout << multipleChoiceLetters[i] << ") " << choiceArray[i] <<
"\n";
    }
    std::cout << "\n";

    std::cout << "Your choice: ";
    char decision;
    std::cin >> decision;
    return decision;
}
char multipleChoicePrompt(std::string choiceA, std::string choiceB,
std::string choiceC, std::string choiceD)
{
    std::string choiceArray[4] = {choiceA, choiceB, choiceC, choiceD};
    for (int i = 0; i <= 3; i++)
    {
        std::cout << multipleChoiceLetters[i] << ") " << choiceArray[i] <<
"\n";
    }
    std::cout << "\n";

    std::cout << "Your choice: ";
    char decision;
    std::cin >> decision;
    return decision;
}
char multipleChoicePrompt(std::string choiceA, std::string choiceB,
std::string choiceC, std::string choiceD, std::string choiceE)
{
    std::string choiceArray[5] = {choiceA, choiceB, choiceC, choiceD,
choiceE};
    for (int i = 0; i <= 4; i++)
    {
        std::cout << multipleChoiceLetters[i] << ") " << choiceArray[i] <<
"\n";
    }
    std::cout << "\n";

    std::cout << "Your choice: ";
```

```
    char decision;
    std::cin >> decision;
    return decision;
}

// Generates a random number between (and including) 2 integers
int randomNumberBetween(int firstNumber, int secondNumber)
{
    int range = secondNumber - firstNumber + 1;
    int randomNumber = (rand() % range) + firstNumber;
    return randomNumber;
}

// Occurs whenever a battle takes place, takes the two characters battling
as arguments
void Battle(Character& character1, Character& character2)
{
    int turn = 1;
    std::cout << "A wild " << character2.name << " approached you to
battle!\n";
    waitFor(3.5);
    system("CLS");

    // Takes turns between the two characters in the battle until one of
them is defeated
    while (character1.hitpoints >= 1 && character2.hitpoints >= 1)
    {
        std::cout << "Turn " << turn << "\n";
        displayHealthbars(character1, character2);

        if (turn % 2 == 1)
        {
            char choice;
            choice = multipleChoicePrompt("Attack", "Heal");
            std::cout << "\n";
            switch (choice)
            {
                // User chooses "Attack"
                case 'A':
                {
```

```
                    characterAttack(character1, character2);
                    break;
                }
                // User chooses "Heal"
                case 'B':
                {
                    characterHeal(character1);
                    break;
                }
                default:
                {
                    std::cout << "Sorry, that is not a valid input.";
                    exit(1);
                }
            }
        }
        else
        {
            characterAttack(character2, character1);
        }
        turn++;
    }

    if (character1.hitpoints <= 0)
    {
        gameOver();
    }
    else
    {
        std::cout << character1.name << " wins!";
        waitFor(3.5);
        system("CLS");
    }
}

// Displays the healthbars of both characters during a battle
void displayHealthbars(Character& character1, Character& character2)
{
    std::cout << character1.name << " Health: \n";
        character1.printHealthbar();
```

```
        std::cout << "\n";
        std::cout << character2.name << " Health: \n";
        character2.printHealthbar();
        std::cout << "\n";
}

// Occurs when any character in a battle attacks
void characterAttack(Character& attacker, Character& defender)
{
    system("CLS");
    std::cout << attacker.name << " attacked!\n";
    waitFor(3.5);
    system("CLS");
    if (randomNumberBetween(1,100) <= attacker.accuracy)
    {
        attacker.attack(&defender);
        std::cout << "The attack landed, and dealt " << attacker.damage <<
" damage!\n";
    }
    else
    {
        std::cout << "The attack missed!\n";
    }
    waitFor(3.5);
    system("CLS");
}

// Occurs when a character heals in a battle
void characterHeal(Character& theCharacter)
{
    int healAmount = 5;
    theCharacter.hitpoints += healAmount;
    int amountHealed;
    if (theCharacter.hitpoints > theCharacter.maxHitpoints)
    {
        amountHealed = healAmount - (theCharacter.hitpoints -
theCharacter.maxHitpoints);
        theCharacter.hitpoints = theCharacter.maxHitpoints;
    }
    else
```

```
    {
        amountHealed = healAmount;
    }

    system("CLS");
    std::cout << theCharacter.name << " healed " << amountHealed << "
hitpoints!\n";
    waitFor(3.5);
    system("CLS");
}
```