University of Brighton
Computer Games

CI411 - Introduction to Game Programming 2022 -23

Coursework 1: Workshop Logs

Module Leader: David Dorrington

Name: Kennedy Sovine

Student number: 22840907

Date: January 7, 2023

Contents

Reflection

This semester, I have learned the fundamentals of C++. I have learned mathematical operators, conditionals, loops, inheritance, arrays, and vectors. These theories are the basics of programming in any language and being able to nail them down allows me to learn more complex coding techniques in the future.

In my experience, readings for coding learning are less valuable than actual practical applications. Just because you read how to do something doesn't give you the skills needed to perform the task correctly. I believe learning is far better than reading and/or theories because, in class, we applied what we learned. In many cases in my coding experience, I would learn how to implement a new skill into a program, but when practiced, it didn't work. The nuances that are encountered when programming cannot be learned through words on a page or a screen.

Honestly, I don't think I could've done better with my approach to problems. I am quite experienced in problem-solving, and one of my greatest strengths is reading comprehension. I could read the problem, analyze it quickly, and then come up with a solution that was not only quick but effective. I often considered cases where a certain input would crash the problem if left unchecked, and I made sure my code was the most effective work I could produce. The only thing I would be able to say I could improve would be styling. My code has very little style, and I would say is rather bulky in some spots.

I think one of the most important things I've learned from this course in making future games would be the usage of vectors. My original coding language I learnt was Java, and one thing I missed from it was array lists. Vectors are just like array lists; in that I can add multiple objects to the list and keep a cohesive and understandable list of all my objects.

I believe my learning in this module is different than how most others may have learned. I already know these basics and fundamentals of coding and I have the experience to know how to approach problems correctly. The only thing I truly learned was how to do what I would do in Java in C++ instead. Even then, the two languages are very similar that it wasn't a learning curve at all. The most I learned was the potential application to use this knowledge for games rather than general software development, which I already knew how to do.

Week:	2	Themes:	*Arithmetic Operators*

Workshop Activity

Images of Source Code and Running Program (Output)

```
Arithmetic Operators
---------------------
10 + 5 = 15

10 - 5 = 5

10 * 5 = 50

10 / 5 = 2

10.0 + 5.0 = 15

11 % 5 = 1

(10 + 5) * 4 = 60

10 + 5 * 4 = 30

---------
Buh bye
```

Image: Output of Arithmetic Operators

```
 Circle Calculations
----------------------
Please enter a value for the radius:  4


Diameter: 8

Circumference: 25.136

Area: 50.272
```

Image: Output of Circle Calculations

```
 ------- Missile Calculation -------
Please input the missile speed (m/s)
13
Please input the missile range (m)
400
Time taken to travel = 30.7692 seconds
```

Image: Output of Missile Calculation

```
------- Character Damage Calc ------
HP = 100

Please input the attack power:
45
Please input the shield power:
30

Damage done: 31.5

Health left: 68.5
```

Image: Character Damage Calc

```
//Math :D
cout << " 10 + 5 = " << 10 + 5 << endl << endl;
cout << " 10 - 5 = " << 10 - 5 << endl << endl;
cout << " 10 * 5 = " << 10 * 5 << endl << endl;

cout << " 10 / 5 = " << 10 / 5 << endl << endl;
cout << " 10.0 + 5.0 = " << 10.0 + 5.0 << endl << endl;
cout << " 11 % 5 = " << 11 % 5 << endl << endl;

cout << " (10 + 5) * 4 = " << (10 + 5) * 4 << endl << endl;
cout << " 10 + 5 * 4 = " << 10 + 5 * 4 << endl << endl;

cout << "---------\n Buh bye" << endl;
```

Image: Arithmetic Operators code

```
    const float PI = 3.142;
    float radius;

    //User Input
    cout << "Please enter a value for the radius:  ";
    cin >> radius;

    //Math
    float diameter = 2 * radius;
    float circumference = 2 * PI * radius;
    float area = PI * (radius * radius);

    //Output
    cout << "\n\nDiameter: " << diameter << endl<<endl;
    cout << "Circumference: " << circumference << endl<<endl;
    cout << "Area: " << area << endl << endl;

    return 0;
}
int rangeMissile() {
    float speed;
    float range;

    cout << "Please input the missile speed (m/s)" << endl;
    cin >> speed;
    cout << "Please input the missile range (m)" << endl;
    cin >> range;

    cout << "Time taken to travel = " << 1.0*(range / speed) << " seconds" << endl << endl << endl;
```

Image: Circle and Missile Range code.

```
    //User Inputs
    float hp = 100.0;
    float ap;
    float sp;

    cout << "HP = "<< hp <<endl<< endl;
    cout << "Please input the attack power: " << endl;
    cin >> ap;

    cout << "Please input the shield power: " << endl;
    cin >> sp;

    //Calculations and output
    hp = hp - (ap * ((100 - sp) / 100));
    float attack = ap * ((100 - sp) / 100);
    cout << "\nDamage done: "<<attack<<endl << endl;
    cout << "Health left: " << hp << endl << endl;
```

Image: Damage Calculator Code

```
cout << " Arithmetic Operators\n----------------------" << endl;
op();

cout << "\n \n Circle Calculations\n----------------------" << endl;
circle();

cout << "\n\n ------- Missile Calculation -------" << endl;
rangeMissile();

cout << "\n\n------- Character Damage Calc ------" << endl;
dmgCalc();
```

Image: Main Method code

Discussion of Workshop Activity

In this week's workshop activity, we learned how to use arithmetic operators in C++. A crucial understanding of math and the order of operations was needed to complete this assessment. Effective and efficient code was developed and properly used. The operators used are: modulus, division, multiplication, addition, subtraction.

Reading

Academic Paper / Book Chapter	

Basic math operators are discussed.

Further Work / Any other Relevant Information

The further work I put in was organizing my code into separate functions rather than having all of it in int main(). I called upon these functions and accurately and effectively used them within main().

Week:	3	Themes:	*Comparative Operators*

Workshop Activity

Images of Source Code and Running Program (Output)

```
int x = 20, y = 18;

if (x == y) {
    cout << "x is equal to y" << endl;
}
else {
    cout << "x is not equal to y" << endl;
}

if (x > y) {
    cout << "x is greater than y" << endl;
}
else {
    cout << "x is less than y" << endl;
}
```

Image: Testing comparative operators

```
int score = 0;

cout << "Enter a score value" << endl;
cin >> score;

if (score > 1000) {
    cout << "Great Score!" << endl;
}
else if (score > 500) {
    cout << "Good Score!" << endl;
}
else {
    cout << "You're score is low" << endl;
}
```

Image: Score Check code

```
int choice = 0;
cout << "=== Option Menu ===" << endl;

cout << "Please Select the program to run" << endl;
cout << " 1: Circle Calculator\n 2: Missle Range Calculator\n 3: Attack Damage Calculator\n 4: Exit Programme" <<
cin >> choice;


if (choice == 1) {
    const float PI = 3.142;
    float radius;

    //User Input
    cout << "Please enter a value for the radius:  ";
    cin >> radius;

    //Math
    float diameter = 2 * radius;
    float circumference = 2 * PI * radius;
    float area = PI * (radius * radius);

    //Output
    cout << "\n\nDiameter: " << diameter << endl << endl;
    cout << "Circumference: " << circumference << endl << endl;
    cout << "Area: " << area << endl << endl;
}

else if (choice == 2) {
    float speed;
    float range;

    cout << "Please input the missile speed (m/s)" << endl;
    cin >> speed;
    cout << "Please input the missile range (m)" << endl;
    cin >> range;

    cout << "Time taken to travel = " << 1.0 * (range / speed) << " seconds" << endl << endl << endl;
}

else if (choice == 3) {
    //User Inputs
    float hp = 100.0;
    float ap;
    float sp;

    cout << "HP = " << hp << endl << endl;
    cout << "Please input the attack power: " << endl;
    cin >> ap;

    cout << "Please input the shield power: " << endl;
    cin >> sp;

    //Calculations and output
    hp = hp - (ap * ((100 - sp) / 100));
    float attack = ap * ((100 - sp) / 100);
    cout << "\nDamage done: " << attack << endl << endl;
    cout << "Health left: " << hp << endl << endl;
}

else {
    exit(0);
}
```

Image: Menu Select Code

```
cout << "==========\n \n \n Character Select and Equip" << endl;

//Variables
string charName;
string weapon;
string race;
int dmg;
int cd;
float atk;
float health = 100.0;
int cSelect;

int ranNum;

string opName;
int opDmg;
int opCd;
float opAtk;
float opHP = 50.0;
float opAtkD;


cout << "\n Please Select a Character" << endl;
cout << "\n 1: Dwarf\n 2: Elf\n 3: Orc\n 4: Fighter\n 5: Wizard\n";
cin >> cSelect;

cout << "What is your name?" << endl;
cin >> charName;

switch (cSelect) {
    //Dwarf
case 1:
    weapon = "a rusty short sword";
    race = "Dwarf";
    dmg = 20;
    cd = 6;
    break;

    //Elf
case 2:
    weapon = "a wooden bow";
    race = "Elf";
    dmg = 16;
    cd = 8;
    break;


    //Orc
case 3:
    weapon = "orcish fists";
    race = "Orc";
    dmg = 15;
    cd = 4;
    break;

    //Fighter
case 4:
    weapon = "brass knuckles";
    race = "Fighter";
    dmg = 17;
    cd = 3;
    break;

    //Wizard
case 5:
    weapon = "~magic~";
    race = "Wizard";
    dmg = 30;
    cd = 9;
    break;

default:
    cout << "You imbecile. You can't even choose properly" << endl;
    exit(0);
}

//Random attack power
srand(time(NULL));
atk = dmg * (rand()%5) * 1 / cd;
```

Image: Character story code 1

```
//Opponent
srand(time(NULL));
opName = "Jeffery the Fisherman";
opDmg = rand() % dmg + 1;
srand(time(NULL));
opCd = rand() % cd + 1;
opAtk = opDmg * 5 * 1 / opCd;

//The Story itself
srand(time(NULL));
ranNum = rand() % 10;

cout << "\n\n ========== The Encounter ==========\n" << endl;
cout << charName << " the " << race << " sets out on a quest to";

switch (ranNum) {
case 1:
    cout << " avenge their dead father ";
    break;
case 2:
    cout << " kill the people who burnt down their hometown ";
    break;
case 3:
    cout << " become stronger because they think they're better than everybody (What a showoff) ";
    break;
default:
    cout << " cause they can ";
}

cout << "with nothing more than " << weapon << endl;
cout <<"On your way to do your task, you run into " << opName << " who just really doesn't like you :/ He attacks!"

//Fight sequence
bool fight = true;
int fChoice;
```

Image: Character Story code 2

```
while (fight) {
    srand(time(NULL));
    cout << "\n\nName: " << charName << "   Attack: " << dmg << "   Health: " << health << endl;
    cout << "===== Action Select =====" << endl;
    cout << " 1: Attack\n 2: Defend" << endl;
    cin >> fChoice;

    //Choice results
    if (fChoice == 1) {
        atk = dmg * (rand() % 5) * 1 / cd;
        if (atk <= 0) {
            atk = 8;
        }

        cout << charName << " the " << race << " attacks with " << weapon << endl;
        cout << "You deal " << atk << " damage" << endl;
        opHP -= atk;
        if (opHP < 0) {
            opHP = 0;
        }
    }
    //Defend
    else if (fChoice == 2) {
        cout << "You defend against an oncoming attack!" << endl;
        srand(time(NULL));
        opAtkD = opAtk - (rand() % 6);
        if (opAtkD < 0) {
            opAtkD = 7;
        }
        cout << opName << " attacks and deals " << opAtkD << " damage" << endl;
        health -= opAtkD;
        continue;
    }
    else {
        continue;
    }

    //Default dmg taken
    cout << opName << " attacks and deals " << opAtk << " damage" << endl;
    health -= opAtk;

    //Change health to 0 if below
    if (health < 0) {
        health = 0;
    }

    //Check to see if fight still going
    if (opHP == 0) {
        fight = false;
        cout << opName << " falls to the ground as you land your final strike.\n You were victorious!" << endl;
    }
    else if (health == 0) {
        fight = false;
        cout << opName << " strikes you down and you fall, leaving your journey forever unfinished.\n You lost." << e
        exit(0);
    }
}

cout << "\nYou continue to the next town" << endl;
cout << "\n========== Game Over ==========";
exit(0);
```

Image: Character Story code 3

```
x is not equal to y
x is greater than y
```

Image: Example output of comparative test

```
Enter a score value
756
Good Score!
```

```
Enter a score value
3000
Great Score!
```

Image: Example output of score check

```
=== Option Menu ===
Please Select the program to run
 1: Circle Calculator
 2: Missle Range Calculator
 3: Attack Damage Calculator
 4: Exit Programme
1
Please enter a value for the radius:  3


Diameter: 6

Circumference: 18.852

Area: 28.278
```

```
=== Option Menu ===
Please Select the program to run
 1: Circle Calculator
 2: Missle Range Calculator
 3: Attack Damage Calculator
 4: Exit Programme
2
Please input the missile speed (m/s)
25
Please input the missile range (m)
10
Time taken to travel = 0.4 seconds
```

```
=== Option Menu ===
Please Select the program to run
 1: Circle Calculator
 2: Missle Range Calculator
 3: Attack Damage Calculator
 4: Exit Programme
3
HP = 100

Please input the attack power:
35
Please input the shield power:
20

Damage done: 28

Health left: 72
```

```
=== Option Menu ===
Please Select the program to run
 1: Circle Calculator
 2: Missle Range Calculator
 3: Attack Damage Calculator
 4: Exit Programme
4

C:\Users\kis16\OneDrive - University of Brighton\KIS16_Modules\CI4
Week3\CI411_Week3\x64\Debug\CI411_Week3.exe (process 20508) exited
Press any key to close this window . . .
```

Images: Output of the menu select

```
==========

 Character Select and Equip

 Please Select a Character

 1: Dwarf
 2: Elf
 3: Orc
 4: Fighter
 5: Wizard
2
What is your name?: Kennedy


 ========== The Encounter ==========

Kennedy the Elf sets out on a quest to cause they can with nothing more than a wooden bow
On your way to do your task, you run into Jeffery the Fisherman who just really doesn't like you :/ He attacks!


Name: Kennedy   Attack: 16   Health: 100
===== Action Select =====
 1: Attack
 2: Defend
1
Kennedy the Elf attacks with a wooden bow
You deal 8 damage
Jeffery the Fisherman attacks and deals 5 damage


Name: Kennedy   Attack: 16   Health: 95
===== Action Select =====
 1: Attack
 2: Defend
2
You defend against an oncoming attack!
Jeffery the Fisherman attacks and deals 1 damage


Name: Kennedy   Attack: 16   Health: 94
===== Action Select =====
 1: Attack
 2: Defend
```

Images: Example output of Character Story code

Discussion of Workshop Activity
In this week's workshop activity, we learned how to use comparative operators in C++ in order to make menus and character story quests. This also included the use of if statements, else if statements, else statements, and switch cases. If statements were used singularly and in nested loops. The comparative operators of addition, subtraction, division, multiplication, and modulus were key to creating the programmes.

Reading	
Academic Paper / Book Chapter	
&& (and) and || (or) operators discussed. Different types of conditional statements in relation to loops.	

Further Work / Any other Relevant Information
The further work I put in was making random sequences happen in my character story. Your motive for traveling changes and the damage you do to the NPC is random every time. The damage done to you by the NPC is also random and based upon your own stats depending on the race you chose. I believe the further work I completed showcased my literacy in C++ and similar coding languages.

Week:	4	Themes:	*Loops*

Workshop Activity

Images of Source Code and Running Program (Output)

```cpp
for (int i = 1; i ≤ 20; i++) {
    cout ≪ i ≪ " ";
}

cout ≪ "\n\n";
for (int i = 100; i ≥ 0; i -= 5) {
    cout ≪ i ≪ " ";
}
```

```cpp
cout ≪ "═══ Times Table ═══" ≪ endl;
int num;
cout ≪ "Input a Number" ≪ endl;
cin ≫ num;

cout ≪ "\n";

for (int row = 1; row ≤ 12; row++) {
    for (int col = 1; col ≤ 12; col++) {
        cout ≪ (num * row) * col ≪ " ";
    }
    cout ≪ endl;
}
```

Image: Code for Task 1

```cpp
cout ≪ "═══ Times Table ═══" ≪ endl;
int num;
char playAgain = 'y';

while (playAgain ≠ 'n') {
    cout ≪ "Input a Number" ≪ endl;
    cin ≫ num;

    cout ≪ "\n";
    for (int row = 1; row ≤ 12; row++) {
        for (int col = 1; col ≤ 12; col++) {
            cout ≪ (num * row) * col ≪ " ";
        }
        cout ≪ endl;
    }

    cout ≪ "\n\nDo you want to go again? (y/n)" ≪ endl;
    cin ≫ playAgain;

    playAgain = tolower(playAgain);

}
```

Image: Code for Task 2

```
cout << "========== Simple Combat Loop ==========\n" << endl;

//conditionals
bool gameOver = false;
int playerChoice;

//player stats
double pcHP = 100.0;
int pcATK = 40;
double pcATKMod;
double newPAtk;

//npc stats
double npcHP = 100.0;
int npcATK = 40;
double npcATKMod;
double newNAtk;
while (!gameOver) {

    //Setup
    srand(time(NULL));
    cout << "\n======= Please Select an Action =======";
    cout << "\n 1: Attack\n 2: Defend\n 3: Evade\n\n Choice: ";
    cin >> playerChoice;

    //Attack Modifier
    switch (playerChoice) {
    case 1:
        pcATKMod = rand() % 6 + 1;
        break;

    case 2:
        pcATKMod = rand() % 5 + 1;
        break;

    case 3:
        pcATKMod = rand() % 4 + 1;
        break;
    default:
        cout << "\n Not a Valid Option";
        pcATKMod = 100;
    }

    npcATKMod = rand() % 5 + 1;

    newNAtk = npcATK * 1 / npcATKMod;
    newPAtk = pcATK * 1 / pcATKMod;

    cout << "\n\nYour Attack = " << int(newNAtk) << endl;
    cout << "NPC Attack = " << int(newPAtk) << endl;

    npcHP -= newPAtk;
    pcHP -= newNAtk;
```

```
    //If HP < 0, HP = 0
    if (pcHP < 0) {
        pcHP = 0;
    }
    if (npcHP < 0) {
        npcHP = 0;
    }

    //Say new HP
    cout << "\n\n\n PC Health : " << int(pcHP) << endl;
    cout << " New NPC Health : " << int(npcHP) << endl;

    if (pcHP == 0 || npcHP == 0) {
        gameOver = true;
    }
}

cout << "\n\n========== GAME OVER ==========";
if (pcHP == 0) {
    cout << "\n           YOU LOST" << endl;
}
else {
    cout << "\n           YOU WON" << endl;
}
```

Images: Code of Task 3

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

100 95 90 85 80 75 70 65 60 55 50 45 40 35 30 25 20 15 10 5 0
```

Image: Output of Task 1

```
====== Times Table =====
Input a Number
1

1 2 3 4 5 6 7 8 9 10 11 12
2 4 6 8 10 12 14 16 18 20 22 24
3 6 9 12 15 18 21 24 27 30 33 36
4 8 12 16 20 24 28 32 36 40 44 48
5 10 15 20 25 30 35 40 45 50 55 60
6 12 18 24 30 36 42 48 54 60 66 72
7 14 21 28 35 42 49 56 63 70 77 84
8 16 24 32 40 48 56 64 72 80 88 96
9 18 27 36 45 54 63 72 81 90 99 108
10 20 30 40 50 60 70 80 90 100 110 120
11 22 33 44 55 66 77 88 99 110 121 132
12 24 36 48 60 72 84 96 108 120 132 144
```

Image: Output of Task 1

```
====== Times Table =====
Input a Number
2

2 4 6 8 10 12 14 16 18 20 22 24
4 8 12 16 20 24 28 32 36 40 44 48
6 12 18 24 30 36 42 48 54 60 66 72
8 16 24 32 40 48 56 64 72 80 88 96
10 20 30 40 50 60 70 80 90 100 110 120
12 24 36 48 60 72 84 96 108 120 132 144
14 28 42 56 70 84 98 112 126 140 154 168
16 32 48 64 80 96 112 128 144 160 176 192
18 36 54 72 90 108 126 144 162 180 198 216
20 40 60 80 100 120 140 160 180 200 220 240
22 44 66 88 110 132 154 176 198 220 242 264
24 48 72 96 120 144 168 192 216 240 264 288


Do you want to go again? (y/n)
n
```

Image: Output of Task 2

```
========== Simple Combat Loop ==========


======= Please Select an Action =======
 1: Attack
 2: Defend
 3: Evade

 Choice: 1


Your Attack = 8
NPC Attack = 8



 PC Health : 92
 New NPC Health : 92

======= Please Select an Action =======
 1: Attack
 2: Defend
 3: Evade

 Choice: 2


Your Attack = 10
NPC Attack = 40



 PC Health : 82
 New NPC Health : 52

======= Please Select an Action =======
 1: Attack
 2: Defend
 3: Evade

 Choice: 3
```

```
======= Please Select an Action =======
 1: Attack
 2: Defend
 3: Evade

 Choice: 1


Your Attack = 20
NPC Attack = 40



 PC Health : 4
 New NPC Health : 0


========== GAME OVER ==========
           YOU WON
```

Image: Output of Task 3

Discussion of Workshop Activity

In this week's workshop activity, we practiced the usage of for and while loops. The code used is like the syntax of java, so the usage of these concepts was not foreign. In task 1, the activity instructed to count to 20 using a for loop and then down from 100, in increments of 5, in a for loop, which was conducted successfully. Task 1 also asked for the implementation of a times table up to 12, which was also done successfully and efficiently.

Task 2 asked to make the times table again, but in a while loop. It had to continue until the user inputted 'N' or 'n' to quit.

Task 3 was like the final task of last week, especially since last weeks went above the requirements. The simple combat loop developed works correctly and efficiently, as demonstrated by the screenshots.

Reading

Academic Paper / Book Chapter	
Simple loop work.	

Further Work / Any other Relevant Information

I looked up some webpages on how to cast doubles as int in C++ in order to output a whole number to the user, so they don't get overwhelmed by decimals. Since it's a cast, it didn't change the calculations, which were still done with doubles.

Week:	5	Themes:	*Functions in C++*

Workshop Activity

Images of Source Code and Running Program (Output)

```
#include <iostream>
using namespace std;

double pi = 3.14;
bool gameOver;

double circleArea(int radius);
double circleCircumference(int radius);
double hypotenuse(double a, double b);


void startScreen();
void endScreen();
void mainMenu();
void forLoopEx();
void combatWhileEx();
void switchRPG();

int main(){
    cout << circleArea(10) << endl;
    cout << circleCircumference(10) << endl;

    cout << hypotenuse(3.0, 4.0) << endl;

    gameOver = false;

    startScreen();

    while (!gameOver) {
        mainMenu();
    }

    endScreen();

}
```

Image: Function declarations, global variables, and the main() function.

```
double circleArea(int radius) {
    double area = pi * (radius * radius);

    return area;
}

double circleCircumference(int radius) {
    double circumference = 2 * pi * radius;

    return circumference;
}

double hypotenuse(double a, double b) {
    a = a * a;
    b = b * b;

    a = (double)sqrt(a);
    b = (double)sqrt(b);

    double c = a + b;

    return c;
}

void startScreen() {
    cout << "Welcome to the Game!" << endl;
}

void endScreen(){
    cout << "\nThanks for playing!";
    exit(0);
}
```

Image: the functions

```
void mainMenu() {

    int select;
    cout << "\n\n========== Option Menu ==========" << endl;
    cout << " 1. Combat Example\n 2. RPG Example \n 3. For Loop\n 4. Exit\n";
    cin >> select;

    switch (select) {
    case 1:
        combatWhileEx();
        break;
    case 2:
        switchRPG();
        break;
    case 3:
        forLoopEx();
        break;
    case 4:
        gameOver = true;
        break;
    default:
        break;
    }
}
```

```
void forLoopEx() {
    for (int i = 1; i <= 20; i++) { ... }

    cout << "\n\n";
    for (int i = 100; i >= 0; i -= 5) { ... }

    cout << "\n====== Times Table ======" << endl;
    int num;
    cout << "Input a Number" << endl;
    cin >> num;

    cout << "\n";

    for (int row = 1; row <= 12; row++) { ... }
}
```

```
void combatWhileEx() {
    cout << "========== Simple Combat Loop ==========\n" << endl;

    //conditionals
    bool gameOver = false;
    int playerChoice;

    //player stats
    double pcHP = 100.0;
    int pcATK = 40;
    double pcATKMod;
    double newPAtk;

    //npc stats
    double npcHP = 100.0;
    int npcATK = 40;
    double npcATKMod;
    double newNAtk;

    while (!gameOver) { ... }

    cout << "\n\n========== GAME OVER ==========";
    if (pcHP == 0) {
        cout << "\n           YOU LOST" << endl;
    }
    else {
        cout << "\n           YOU WON" << endl;
    }
}
```

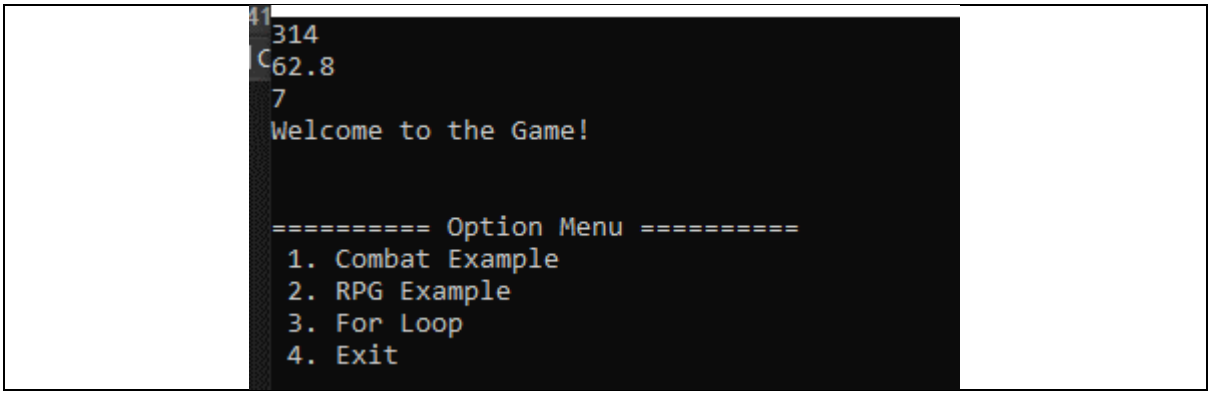
```
void switchRPG() {
    cout << "==========\n \n Character Select and Equip" << endl;

    //Variables
    string charName;
    string weapon;
    string race;
    int dmg;
    int cd;
    float atk;
    float health = 100.0;
    int cSelect;

    int ranNum;

    string opName;
    int opDmg;
    int opCd;
    float opAtk;
    float opHP = 50.0;
    float opAtkD;


    cout << "\n Please Select a Character" << endl;
    cout << "\n 1: Dwarf\n 2: Elf\n 3: Orc\n 4: Fighter\n 5: Wizard\n";
    cin >> cSelect;

    cout << "What is your name?: ";
    cin >> charName;

    switch (cSelect) {
        //Dwarf
    case 1:
        weapon = "a rusty short sword";
        race = "Dwarf";
        dmg = 20;
        cd = 6;
        break;

        //Elf
    case 2:
        weapon = "a wooden bow";
        race = "Elf";
        dmg = 16;
        cd = 8;
        break;
```

```
case 3:
    weapon = "orcish fists";
    race = "Orc";
    dmg = 15;
    cd = 4;
    break;

    //Fighter
case 4:
    weapon = "brass knuckles";
    race = "Fighter";
    dmg = 17;
    cd = 3;
    break;

    //Wizard
case 5:
    weapon = "~magic~";
    race = "Wizard";
    dmg = 30;
    cd = 9;
    break;

default:
    cout << "You imbecile. You can't even choose properly" << endl;
    exit(0);
}

//Random attack power
srand(time(NULL));
atk = dmg * (rand() % 5) * 1 / cd;

//Opponent
srand(time(NULL));
opName = "Jeffery the Fisherman";
opDmg = rand() % dmg + 1;
srand(time(NULL));
opCd = rand() % cd + 1;
opAtk = opDmg * 5 * 1 / opCd;

//The Story itself
srand(time(NULL));
ranNum = rand() % 10;

cout << "\n\n ========== The Encounter ==========\n" << endl;
cout << charName << " the " << race << " sets out on a quest to";
```

```
case 3:
    weapon = "orcish fists";
    race = "Orc";
    dmg = 15;
    cd = 4;
    break;

    //Fighter
case 4:
    weapon = "brass knuckles";
    race = "Fighter";
    dmg = 17;
    cd = 3;
    break;

    //Wizard
case 5:
    weapon = "~magic~";
    race = "Wizard";
    dmg = 30;
    cd = 9;
    break;

default:
    cout << "You imbecile. You can't even choose properly" << endl;
    exit(0);
}

//Random attack power
srand(time(NULL));
atk = dmg * (rand() % 5) * 1 / cd;

//Opponent
srand(time(NULL));
opName = "Jeffery the Fisherman";
opDmg = rand() % dmg + 1;
srand(time(NULL));
opCd = rand() % cd + 1;
opAtk = opDmg * 5 * 1 / opCd;

//The Story itself
srand(time(NULL));
ranNum = rand() % 10;

cout << "\n\n ========== The Encounter ==========\n" << endl;
cout << charName << " the " << race << " sets out on a quest to";
```

```
switch (ranNum) {
case 1:
    cout << " avenge their dead father ";
    break;
case 2:
    cout << " kill the people who burnt down their hometown ";
    break;
case 3:
    cout << " become stronger because they think they're better than everybody (What a showoff) ";
    break;
default:
    cout << " cause they can ";
}

cout << "with nothing more than " << weapon << endl;
cout << "On your way to do your task, you run into " << opName << " who just really doesn't like you :/ He attacks!" << endl;

//Fight sequence
bool fight = true;
int fChoice;

while (fight) {
    srand(time(NULL));
    cout << "\n\nName: " << charName << "   Attack: " << dmg << "   Health: " << health << endl;
    cout << "===== Action Select =====" << endl;
    cout << " 1: Attack\n 2: Defend" << endl;
    cin >> fChoice;

    //Choice results
    if (fChoice == 1) {
        atk = dmg * (rand() % 5) * 1 / cd;
        if (atk <= 0) {
            atk = 8;
        }

        cout << charName << " the " << race << " attacks with " << weapon << endl;
        cout << "You deal " << atk << " damage" << endl;
        opHP -= atk;
        if (opHP < 0) {
            opHP = 0;
        }
    }
    //Defend
    else if (fChoice == 2) {
        cout << "You defend against an oncoming attack!" << endl;
        srand(time(NULL));
        opAtkD = opAtk - (rand() % 6);
        if (opAtkD < 0) {
            opAtkD = 7;
        }
        cout << opName << " attacks and deals " << opAtkD << " damage" << endl;
        health -= opAtkD;
        continue;
    }
    else {
        continue;
    }

    //Default dmg taken
    cout << opName << " attacks and deals " << opAtk << " damage" << endl;
    health -= opAtk;

    //Change health to 0 if below
    if (health < 0) {
        health = 0;
    }

    //Check to see if fight still going
    if (opHP == 0) {
        fight = false;
        cout << opName << " falls to the ground as you land your final strike.\n You were victorious!" << endl;
    }
    else if (health == 0) {
        fight = false;
        cout << opName << " strikes you down and you fall, leaving your journey forever unfinished.\n You lost." << endl;
    }
}

cout << "\nYou continue to the next town" << endl;
cout << "\n========== Game Over ==========";
```

Images: more code


314
62.8
7
Welcome to the Game!
========== Option Menu ==========
1. Combat Example
2. RPG Example
3. For Loop
4. Exit

```
314
62.8
7
Welcome to the Game!


========== Option Menu ==========
 1. Combat Example
 2. RPG Example
 3. For Loop
 4. Exit
1
========== Simple Combat Loop ==========


======= Please Select an Action =======
 1: Attack
 2: Defend
 3: Evade

 Choice: 1


Your Attack = 40
NPC Attack = 8



 PC Health : 60
 New NPC Health : 92

======= Please Select an Action =======
 1: Attack
 2: Defend
 3: Evade

 Choice: 1


Your Attack = 10
NPC Attack = 6



 PC Health : 50
 New NPC Health : 85

======= Please Select an Action =======
 1: Attack
 2: Defend
 3: Evade

 Choice: 1


Your Attack = 10
NPC Attack = 6



 PC Health : 40
 New NPC Health : 78
```

```
========== Option Menu ==========
 1. Combat Example
 2. RPG Example
 3. For Loop
 4. Exit
2
==========

 Character Select and Equip

 Please Select a Character

 1: Dwarf
 2: Elf
 3: Orc
 4: Fighter
 5: Wizard
1
What is your name?: E


 ========== The Encounter ==========

E the Dwarf sets out on a quest to cause they can with nothing more than a rusty short sword
On your way to do your task, you run into Jeffery the Fisherman who just really doesn't like you :/ He attacks!


Name: E   Attack: 20   Health: 100
===== Action Select =====
 1: Attack
 2: Defend
1
E the Dwarf attacks with a rusty short sword
You deal 8 damage
Jeffery the Fisherman attacks and deals 55 damage


Name: E   Attack: 20   Health: 45
===== Action Select =====
 1: Attack
 2: Defend
1
E the Dwarf attacks with a rusty short sword
You deal 3 damage
Jeffery the Fisherman attacks and deals 55 damage
Jeffery the Fisherman strikes you down and you fall, leaving your journey forever unfinished.
 You lost.

You continue to the next town

========== Game Over ==========
```

```
========== Option Menu ==========
 1. Combat Example
 2. RPG Example
 3. For Loop
 4. Exit
3
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

100 95 90 85 80 75 70 65 60 55 50 45 40 35 30 25 20 15 10 5 0
====== Times Table =====
Input a Number
1

1 2 3 4 5 6 7 8 9 10 11 12
2 4 6 8 10 12 14 16 18 20 22 24
3 6 9 12 15 18 21 24 27 30 33 36
4 8 12 16 20 24 28 32 36 40 44 48
5 10 15 20 25 30 35 40 45 50 55 60
6 12 18 24 30 36 42 48 54 60 66 72
7 14 21 28 35 42 49 56 63 70 77 84
8 16 24 32 40 48 56 64 72 80 88 96
9 18 27 36 45 54 63 72 81 90 99 108
10 20 30 40 50 60 70 80 90 100 110 120
11 22 33 44 55 66 77 88 99 110 121 132
12 24 36 48 60 72 84 96 108 120 132 144
```

```
========== Option Menu ==========
 1. Combat Example
 2. RPG Example
 3. For Loop
 4. Exit
4

Thanks for playing!
```

Images: Code Output

Discussion of Workshop Activity

In this week's workshop activity, we were instructed to use functions to write our code outside of the main() program to improve the functionality and readability of the code. We re-introduced programs made in previous weeks to make a main menu that has a start screen and end screen, all called upon in main. The menu and all previous programs work perfectly as expected.

Reading

Academic Paper / Book Chapter	

Functions help make code easier to read.

Further Work / Any other Relevant Information

I also used global variables such as pi and gameOver as to eliminate the need to declare and initialize them in various functions. This allows me to modify the quality of gameOver easily between functions.

Week:	7	**Themes:**	*Arrays and GameBoards*

Workshop Activity

Images of Source Code and Running Program (Output)

```
// CI411_Week7.cpp
// Kennedy Sovine UoB
//

#include <iostream>
using namespace std;

//Functions

//WorkShop Tasks
void displayDaysOfWeek();
void displayPlayArea();

//Further Work
void createBoard();
void displayBoard();
void randomObjectsOnBoard();

//Global
bool gameOver = false;

const int boardHeight = 10;
const int boardWidth = 10;
char gameBoard[boardHeight][boardWidth];
```

Image: Function declarations and further work

```
int main() {

    cout << "====Days of the Week====" << endl;
    displayDaysOfWeek();
    cout << "\n";

    cout << "====Play Area====" << endl;
    displayPlayArea();
    cout << "\n";

    cout << "====Game Board====" << endl;
    createBoard();
    randomObjectsOnBoard();
    displayBoard();

    //Start Screen
    //Set Up
    //Main Loop
    //End Screen

    return 0;
}
```

Image: Main function

```
//Functions
void displayDaysOfWeek() {
    string days[7] = { "Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday" };

    for (int i = 1; i <= 7; i++) {
        cout << "Day " << i << " : " << days[i - 1] << endl;
    }
}
```

Image: Days of the Week

```
void displayPlayArea() {

    //Make play area
    char playArea[8][8] = { {'=','=','=','=','=','=','=', '='},
                            {'|','-','-','-','-','-','-','|'},
                            {'|','-','-','-','-','-','-','|'},
                            {'|','-','-','-','-','-','-','|'},
                            {'|','-','-','-','-','-','-','|'},
                            {'|','-','-','-','-','-','-','|'},
                            {'|','-','-','-','-','-','-','|'},
                            {'=','=','=','=','=','=','=','='}};

    //Random Populace
    srand(_Seed:time(_Time:NULL));

    for (int i = 0; i < 5; i++) {

        int row = (rand() % (6) + 1);
        int col = (rand() % (6)) + 1;

        playArea[row][col] = '*';
    }


    //Display play area

    for (int row = 0; row < 8; row++) {
        for (int col = 0; col < 8; col++) {
            cout << playArea[row][col];
            cout << " ";
        }
        cout << "\n";
    }
}
```

Image: Task2, display play area.

```
void createBoard() {
    for (int row = 0; row < boardHeight; row++) {
        for (int col = 0; col < boardWidth; col++) {

            if (row == 0 || row == boardHeight - 1) {
                gameBoard[row][col] = '=';
            }
            else {
                if (col == 0 || col == boardWidth - 1) {
                    gameBoard[row][col] = '|';
                }
                else {
                    gameBoard[row][col] = '-';
                }
            }
        }
    }
}

void displayBoard() {

    for (int row = 0; row < boardHeight; row++) {
        for (int col = 0; col < boardWidth; col++) {
            cout << gameBoard[row][col];
            cout << " ";
        }
        cout << "\n";
    }

}
```

Image: Further work, create and display board

```
void randomObjectsOnBoard() {

    char randObjects[5] = { 'A', 'M', 'G', 'E', 'L' };
    srand(_Seed:time(_Time:NULL));

    for (int i = 0; i < 5; i++) {

        int row = (rand() % (boardHeight - 2) + 1);
        int col = (rand() % (boardWidth - 2)) + 1;

        if (gameBoard[row][col] != '-') {
            i--;
        }
        else {
            gameBoard[row][col] = randObjects[i];
        }
    }
}
```

Image: Further work, random objects on the gameboard

```
====Days of the Week====
Day 1 : Sunday
Day 2 : Monday
Day 3 : Tuesday
Day 4 : Wednesday
Day 5 : Thursday
Day 6 : Friday
Day 7 : Saturday
```

```
====Play Area====
= = = = = = = = =
| - - - - - - - |
| * - - - - - - |
| - - - - - - - |
| - - - - * - - |
| * - - - * * - |
| - - - - - - - |
= = = = = = = = =
```

```
====Game Board====
= = = = = = = = = = =
| - - M - - - - - - |
| - - - - - - - - - |
| - - - - - - L - - |
| - - - - - - - - - |
| - - - - - - - - - |
| E - - - - - - - - |
| - - - - - - - - A |
| - - - - G - - - - |
= = = = = = = = = = =
```

Images: Output of all code

Discussion of Workshop Activity

In this weeks' workshop activity, we learned how to create and fill arrays along with iterating through them using for loops and for each loops.

Reading

Academic Paper / Book Chapter	
Array work for game boards.	

Further Work / Any other Relevant Information
In my further work of inputting random letters onto the game board, I made an array with random letters inside and created a loop that filled a random section of the board with the letter. It also included a catch statement that checked if there was a letter already placed in that spot. If that was the case, 'i' would decrease by 1 and it would go backwards into the program again, therefore assuring that there will always be 5 random characters on the board.

Week:	8	Themes:	*Classes and Objects*

Workshop Activity

Images of Source Code and Running Program (Output)

```
// ======================================================
// CI411_Week8
// Kennedy Sovine UoBGames, 23 November 2022
// ======================================================

// Libraries to include
#include <iostream>
using namespace std;

// ======================================================
// Function Declaration

int getAtr(int max);

// ======================================================
// Classes
class Circle {
public:
    float radius;
    double getArea() {
        return pi * (radius * radius);
    }
    double getCircumference() {
        return 2 * pi * radius;
    }
private:
    double pi = 3.14;
};
```

Image: code with circle class

```
class Square {
    public:
        float sideLength;
        double getArea() {
            return sideLength * sideLength;
        }
        double getPerimeter() {
            return sideLength + sideLength + sideLength + sideLength;
        }
};
```

Image: code with square class

```
class Character {
    public:
        int health, speed, stamina, strength;
        int xp;
        string name, weapon;
        bool armed, isAlive;
        //Creates PC
        Character(string cname, string cweapon) {
            name = cname;
            weapon = cweapon;
            health = 100;
            speed = getAtr(10);
            stamina = getAtr(10);
            strength = getAtr(10);
            xp = getAtr(100);
            armed = true;
            isAlive = true;
        }
        Character(string cname) {
            name = cname;
            weapon = "";
            health = 100;
            speed = getAtr(10);
            stamina = getAtr(10);
            strength = getAtr(10);
            xp = getAtr(100);
            armed = false;
            isAlive = true;
        }
        //Creates NPC
        Character() {
            name = "NPC";
            health = 100;
            speed = getAtr(5);
            stamina = getAtr(5);
            strength = getAtr(5);
            xp = 0;
            armed= true;
            isAlive = true;
        }
```

Image: code with Character Class

```
int main() {
    cout << "===== Circles =====" << endl;
    Circle circle1;
    circle1.radius = 2;

    Circle circle2;
    circle2.radius = 4;

    cout << circle1.getArea() << endl;
    cout << circle2.getCircumference() << endl;

    cout << "===== Squares =====" << endl;

    Square square1;
    square1.sideLength = 2;

    Square square2;
    square2.sideLength = 4;

    cout << square1.getArea() << endl;
    cout << square2.getPerimeter() << endl;

    cout << "===== Characters =====" << endl;
    Character PC1("Kennedy", "Bow and Arrow");
    srand(time(NULL));
    Character PC2("Harry");

    srand(time(NULL));
    Character NPC1;

    cout << "Player Character 1: " + PC1.name << endl;
    cout << "HP: "; cout << PC1.health;
    cout << " Stamina: "; cout << PC1.stamina;
    cout << " Speed: "; cout << PC1.speed;
    cout << " XP: "; cout << PC1.xp << endl;
    cout << "Alive = "; cout << PC1.isAlive;
    cout << " : Armed = "; cout << PC1.armed;
```

Image: main with circle, square and character being used

```
    cout << "\n\n";

    cout << "Player Character 2: " + PC2.name << endl;
    cout << "HP: "; cout << PC2.health;
    cout << " Stamina: "; cout << PC2.stamina;
    cout << " Speed: "; cout << PC2.speed;
    cout << " XP: "; cout << PC2.xp << endl;
    cout << "Alive = "; cout << PC2.isAlive;
    cout << " : Armed = "; cout << PC2.armed;

    cout << "\n\n\n";

    cout << "NPC: " + NPC1.name << endl;
    cout << "HP: "; cout << NPC1.health;
    cout << " Stamina: "; cout << NPC1.stamina;
    cout << " Speed: "; cout << NPC1.speed;
    cout << " XP: "; cout << NPC1.xp << endl;
    cout << "Alive = "; cout << NPC1.isAlive;
    cout << " : Armed = "; cout << NPC1.armed;

    cout << "\n";




    return 0;
}

// ========================================================
// Functions

int getAtr(int max) {
    return (rand() % max);
}

// ========================================================
```

Image: Code that shows getAtr() and the rest of the code for the main class.

```
===== Circles =====
12.56
25.12
===== Squares =====
4
16
===== Characters =====
Player Character 1: Kennedy
HP: 100 Stamina: 7 Speed: 1 XP: 0
Alive = 1 : Armed = 1

Player Character 2: Harry
HP: 100 Stamina: 4 Speed: 5 XP: 19
Alive = 1 : Armed = 0


NPC: NPC
HP: 100 Stamina: 4 Speed: 0 XP: 0
Alive = 1 : Armed = 1
```

Image: output from every task

Discussion of Workshop Activity

In this week's workshop, we learned how to use classes and objects. We had to make the object and then declare its attributes. There also were private and public variables inside classes which change whether or not functions such as main have access to them.

Reading

Academic Paper / Book Chapter	

Classes and objects can make characters and store information to be used later on and continuously updated.

Further Work / Any other Relevant Information

I went further and used constructors and I attempted to make n list or vector to auto iterate through the character objects no matter how many there were, but it failed.

Week:	9	Themes:	*Classes and Inheritance*

Workshop Activity
Images of Source Code and Running Program (Output)

```
Character has been spawned! Character has been spawned! Stats for:
----------
Health: 100 Speed: 5 Alive: 1

Stats for:
----------
Health: 0 Speed: 5 Alive: 0

Character has been spawned! Name: player3 Health: 80

Character has been spawned! Name: player4 Health: 97

Character has been spawned! An NPC from the water faction has spawned!
Character has been spawned! A PC has been spawned! Name: PC1 Health: 100 Mana: 20 Potion: Mana
```

Image: Output from all code

```
// ============================================================
// CI411_Week9
// Kennedy Sovine UoBGames, 30/11/2022
// ============================================================
//
// Libraries to include
#include <iostream>
using namespace std;

// ============================================================
// Function Declaration

// ============================================================
// Classes

class Character {
public:
    string name;
    float health, speed;
    bool isAlive;

    void checkIsAlive() {
        if (health <= 0) {
            isAlive = false;
        }
    }
    void displayStats();

    Character() {
        name = "";
        health = 100;
        speed = 5;
        isAlive = true;

        cout << "Character has been spawned! ";
    }

    Character(string nameSet, float healthSet);
};

class NPC : public Character {
public:
    string faction, message;

    NPC(string factionSet);
};

class PC : public Character {
public:
    string potionCarrying;
    int mana;

    void displayStats();

    PC(string nameSet, float healthSet, string potionSet, int manaSet);
};
```

Image: Classes

```
// ========================================
int main(){

    Character player1;
    Character player2;

    player2.health = 0;

    player1.checkIsAlive();
    player2.checkIsAlive();

    player1.displayStats();
    player2.displayStats();

    Character player3("player3", 80);
    Character player4("player4", 97);

    NPC NPC1("water");
    PC PC1("PC1", 100, "Mana", 20);

    return 0;
}
```

Image: main function

```
// ========================================
// Functions & Class Methods

void Character::displayStats(){ ... }

void PC::displayStats(){ ... }

Character::Character(string nameSet, float healthSet) {
    name = nameSet;
    health = healthSet;
    speed = 5;
    isAlive = true;

    cout << "Character has been spawned! ";
    cout << "Name: " << name << " Health: " << health << endl << endl;;
}

NPC::NPC(string factionSet) {
    if (factionSet == "water") {
        faction = "water";
        message = "An NPC from the water faction has spawned! ";
    }
    else if (factionSet == "fire") {
        faction = "fire";
        message = "An NPC from the fire faction has spawned! ";
    }
    else if (factionSet == "earth") {
        faction = "earth";
        message = "An NPC from the earth faction has spawned! ";

    }
    else if (factionSet == "air") {
        faction = "air";
        message = "An NPC from the air faction has spawned! ";
    }
    else {
        faction = "nomad";
        message = "An NPC without a faction has spawned! ";
    }
    cout << message << endl;
}

PC::PC(string nameSet, float healthSet, string potionSet, int manaSet) {
    name = nameSet;
    health = healthSet;
    potionCarrying = potionSet;
    mana = manaSet;
    speed = 5;
    isAlive = true;

    cout << "A PC has been spawned! ";
    cout << "Name: " << name << " Health: " << health << " Mana: " << mana << " Potion: " << potionCarrying << endl << endl;
}

// ========================================
```

Image: Functions and Class Methods

Discussion of Workshop Activity

In this week's workshop activity, we worked on more object usage. We worked with inheritance, as seen with the NPC and PC classes that are derived from the Character class. We also worked with methods and constructors defined inside and outside the class itself.

Reading

Academic Paper / Book Chapter	
Inheritance across files.	
Further Work / Any other Relevant Information	
No further work completed this week.	

Week:	10	Themes:	*Header Files*

Workshop Activity

Images of Source Code and Running Program (Output)

```
// ============================================================
// CI411_Week10
// Kennedy Sovine UoBGames, 7/12/2022
// ============================================================
//
// Libraries to include
#include <iostream>
using namespace std;
#include "GameObjects.h"

// ============================================================
// Function Declaration

// ============================================================
// Global Variables

PC PC1("Geoffry", 100, "none", 50);
NPC NPC1("water");
Character Character1("Bob", 100);

// ============================================================
int main() {

    cout << "\nCharacter Stats: \n\n";

    PC1.displayStats();
    NPC1.displayStats();
    Character1.displayStats();

    return 0;
}
// ============================================================
// Functions & Class Methods


// ============================================================
```

Image: main C++ file

```cpp
#include <iostream>
using namespace std;
#include "GameObjects.h"

void Character::displayStats() {
    cout << "Stats for: " << name << endl;
    cout << "----------" << endl;
    cout << "Health: " << (int)health << " Speed: " << (int)speed << " Alive: " << isAlive << endl << endl;;
}

void PC::displayStats() {
    cout << "Stats for: " << name << endl;
    cout << "----------" << endl;
    cout << "Health: " << (int)health << " Mana: " << mana << " Speed: " << (int)speed << " Potion Carrrying: " << potionCarrying << " Alive: " << isAlive << endl << endl;;
}

void NPC::displayStats() {
    cout << "Stats for: " << name << endl;
    cout << "----------" << endl;
    cout << "Health: " << (int)health << " Faction: " << faction << endl << endl;;
}

Character::Character(string nameSet, float healthSet) {
    name = nameSet;
    health = healthSet;
    speed = 5;
    isAlive = true;

    cout << "Character has been spawned! ";
    cout << "Name: " << name << " Health: " << health << endl << endl;;
}
```

Image: GameObjects.cpp (1)

```cpp
NPC::NPC(string factionSet) {
    if (factionSet == "water") {
        faction = "water";
        message = "An NPC from the water faction has spawned! ";
    }
    else if (factionSet == "fire") {
        faction = "fire";
        message = "An NPC from the fire faction has spawned! ";
    }
    else if (factionSet == "earth") {
        faction = "earth";
        message = "An NPC from the earth faction has spawned! ";

    }
    else if (factionSet == "air") {
        faction = "air";
        message = "An NPC from the air faction has spawned! ";
    }
    else {
        faction = "nomad";
        message = "An NPC without a faction has spawned! ";
    }
    cout << message << endl;
}

PC::PC(string nameSet, float healthSet, string potionSet, int manaSet) {
    name = nameSet;
    health = healthSet;
    potionCarrying = potionSet;
    mana = manaSet;
    speed = 5;
    isAlive = true;

    cout << "A PC has been spawned! ";
    cout << "Name: " << name << " Health: " << health << " Mana: " << mana << " Potion: " << potionCarrying << endl << endl;
}
```

Image: GameObjects.cpp (2)

```h
#pragma once
#include <iostream>
using namespace std;

class Character {
public:
    string name;
    float health, speed;
    bool isAlive;

    void checkIsAlive() {
        if (health <= 0) {
            isAlive = false;
        }
    }
    void displayStats();

    Character() {
        name = "";
        health = 100;
        speed = 5;
        isAlive = true;

        cout << "Character has been spawned! ";
    }

    Character(string nameSet, float healthSet);
};

class NPC : public Character {
public:
    string faction, message;

    NPC(string factionSet);
    void displayStats();
};

class PC : public Character {
public:
    string potionCarrying;
    int mana;

    void displayStats();

    PC(string nameSet, float healthSet, string potionSet, int manaSet);
};
```

Image: GameObjects.h

```
Character has been spawned! A PC has been spawned! Name: Geoffry Health: 100 Mana: 50 Potion: none

Character has been spawned! An NPC from the water faction has spawned!
Character has been spawned! Name: Bob Health: 100


Character Stats:

Stats for: Geoffry
----------
Health: 100 Mana: 50 Speed: 5 Potion Carrrying: none Alive: 1

Stats for:
----------
Health: 100 Faction: water

Stats for: Bob
----------
Health: 100 Speed: 5 Alive: 1
```

Image: Terminal output

Discussion of Workshop Activity

In this workshop activity, all we did was separate the code into 2 new files: A GameObjects.cpp and GameObjects.h .We had to include the header file into the two C++ files so it can reference it.

Reading

Academic Paper / Book Chapter	

Easier to read and better organized code.

Further Work / Any other Relevant Information

References