Odd Semester (2018)



**BINUS UNIVERSITY**

**BINUS INTERNATIONAL**

**Assignment Cover Letter**

**(Group Work****)**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | |  | |  | |
| **Student information:**  **1.Krishita Sukhani 2101716773**  **2. Dean Jourdan Dumais 2101705164**  **3.Ryan Divas Tjahya 2101704413** | | | | |  | |  | |
|  |  |  |  | |
| **Course Code** | **: COMP6340** |  |  | | **Course Name** | | **: Analysis of Algorithms** | |
| **Class** | **: L3BC-BLK** |  |  | | **Name of Lecturer(s)** | | **:** 1. **MARIA SERAPHINA ASTRIANI** | |
|  |  |  |  | |  | | 2. **VINCENT ALEXANDER SELIANG** | |
| **Major** | **: CS** |  |  | |  | |  | |
| **Title of Assignment**  (if any) | : Navigation system using A\* Algorithm and Flood fill algorithm | |  |  | |  | |  | |
| **Type of Assignment**    **Submission Pattern** | **: Final Project** |  |  | |  | |  | |
| **Due Date** | **: 15-1-2019** |  |  | | **Submission Date** | | **: 15-1-2019** | |

**“Navigation system”**

1. **Description**

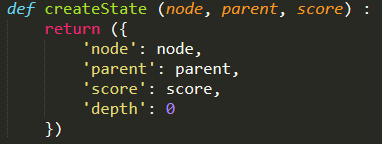
**The function of this program:**

This program is meant to help people find the fastest route towards their destination as well as depicting how A\* Algorithm works and its difference with Flood fill Algorithm

**II. Explanation of Each file**

**A\* Algorithm : ( *Astar.py* )**

**Outside the class**



helper function to create a new state

**Inside the class**

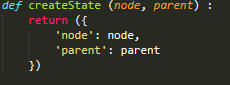
* Finds path with A\*
* Gets next top state from the priority queue
* Gets the current node's index
* Adds current state to the closed state
* Adds state index to do not check
* Gets current state's node neighbor
* Adds new state to the open priority queue
* Sorts the open stack

**Entity: ( *entity.py* )**

* Basically for the green dot that moves from one destination to another
* In charge of the target and speed of the dot

**Floodfill Algorithm: (***Floodfill.py*)

**Outside the class**



Helper function to compute new state

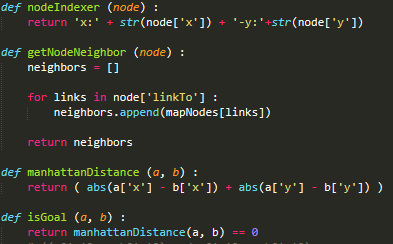
**Inside the class**

* Outside the class
* Get next top state from the stack
* Get the current node's index
* Increment node search count
* Add current state to the closed state
* Add state index to do not check
* Get current state's node neighbor
* Add new state to the open stack

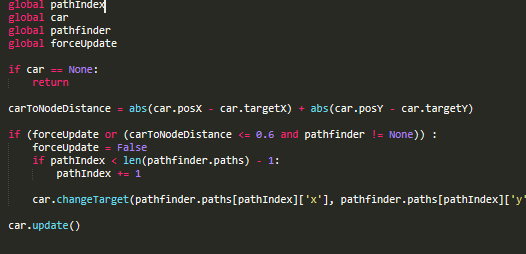
**Pathfinder: (***pathfinder.py*)

* basically loads the astar and flood fill module so that to run the pathfinder function, you only need to call 1 command
* Allows you to easily swap pathfinding algorithm and store the path that the algorithm found

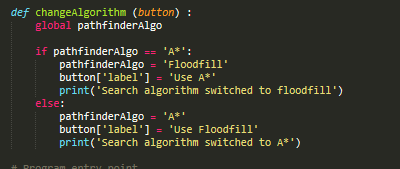
**Pathfinder main: (***pathfinder\_main.py*)



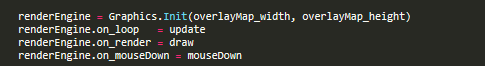
Pathfinder helper functions



Makes the car follow the path



This function allows the user to change between the default A\* algorithm to floodfill algorithm.



This will render the car

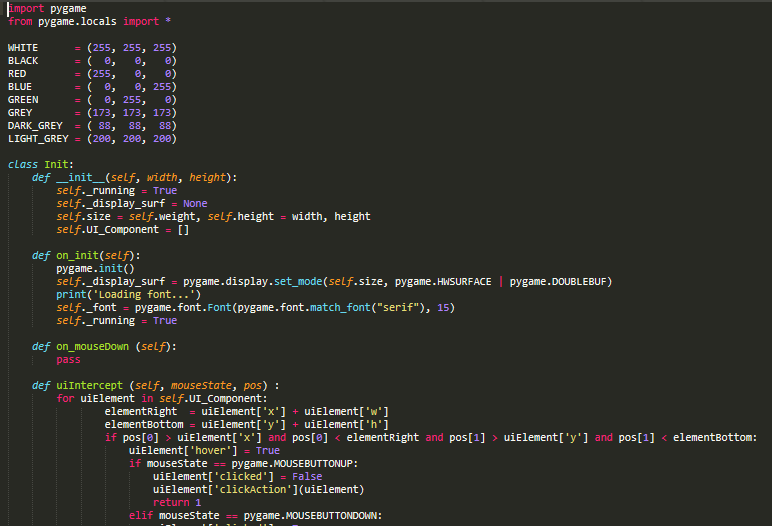


This overlays the map



This is for the button

**Graphics: (***Graphics.py*)



This file consist of the graphics that makes up the maps and also the car movements and lastly the user interface buttons that helps users switch from a\* to floodfil

**Depreciation Menu: (*depreciation menu. h*)**

* Declaration of *void exitchoice()* because the exit option is needed, so it needs to be declared beforehand.
* *void deprecitationChoices()* ask the user to input the choice whether the user wants to go to the compound interest, the single interest, or go back to the main menu.
* After the input, the program will continue to check the condition. To check the user’s input, this project used the while loop to check whether the user input number less than one or more than three.
* After being validate, if the input is deprecitation good value option, user is asked to input the capital, month, and the percentage, then the input will be transferred to the depreciation good value constructor and the constructor will run. Otherwise, if the input is depreciation book value option, user is asked to input the capital, month, and the percentage, then the input will be transferred to the depreciation book value constructor and the constructor will run. Else if the user input the input is go back to the main menu, the *return* statement will help us to return to the main menu in *main.cpp*.
* There are try and catch statement below each of the depreciation constructor. The try block purpose is to check the if condition in each of the get depreciation function (*getDepreciationGoodValue()* to check the good depreciation and *getDepreciationBoodValue()* to check the book depreciation) then if the condition makes exception is thrown, the function would be terminated and the catch block would handle the exception and run the statement inside the catch block if any.
* After the constructor run ( calculated the depreciation according to the formula in each of the constructor and showing the output ), the *exitchoice()* functions are placed after the constructor, so it will asked whether the user wish to exit or to continue the program. If the user wishes to continue, it will return to the interest and depreciation menu that is looped (located in: *main.cpp* ).

***Depreciation.h:***

* Contains of a class named Depreciation that inherit IntAndDep class.
* Declaration of class member functions, constructor, and attribute (class specification):

1. Public:

* Depreciation(double p, double c, double m);
* double getDepreciationGoodValue ()const;
* double getDepreciationBookValue ()const;

***Depreciation.cpp:***

* Contains the function definitions and the constructor definition (class implementation):

1. Depreciation(double p, double c, double m), constructor to set the private member value using the user’s input.
2. double getDepreciationGoodValue ()const, calculate the depreciation good value using the private members (inherited from IntAndDep class) that have been set by the constructor, outputting the value for each month and throwing an exception if the value reach zero or below zero.
3. double getDepreciationBookValue ()const, calculate the depreciation good value using the private members (inherited from IntAndDep class) that have been set by the constructor, outputting the value for each month and throwing an exception if the value reach zero or below zero.

**IntAndDep class: (*IntAndDep.h*)**

* Contains a class named IntAndDep, that consist of:

1. Protected member:

* percent;
* capital;
* month;

1. Public member:

* double getPercent()const, return the value of percent.
* double getCapital()const, return the value of capital.
* double getMonth()const, return the value of month.

**UML Diagram:**

|  |
| --- |
| IntAndDep |
| **#** percent |
| **#** capital |
| **#** month |
| + getPercent()const:double |
| + getCapital()const:double |
| + getMonth()const:double |

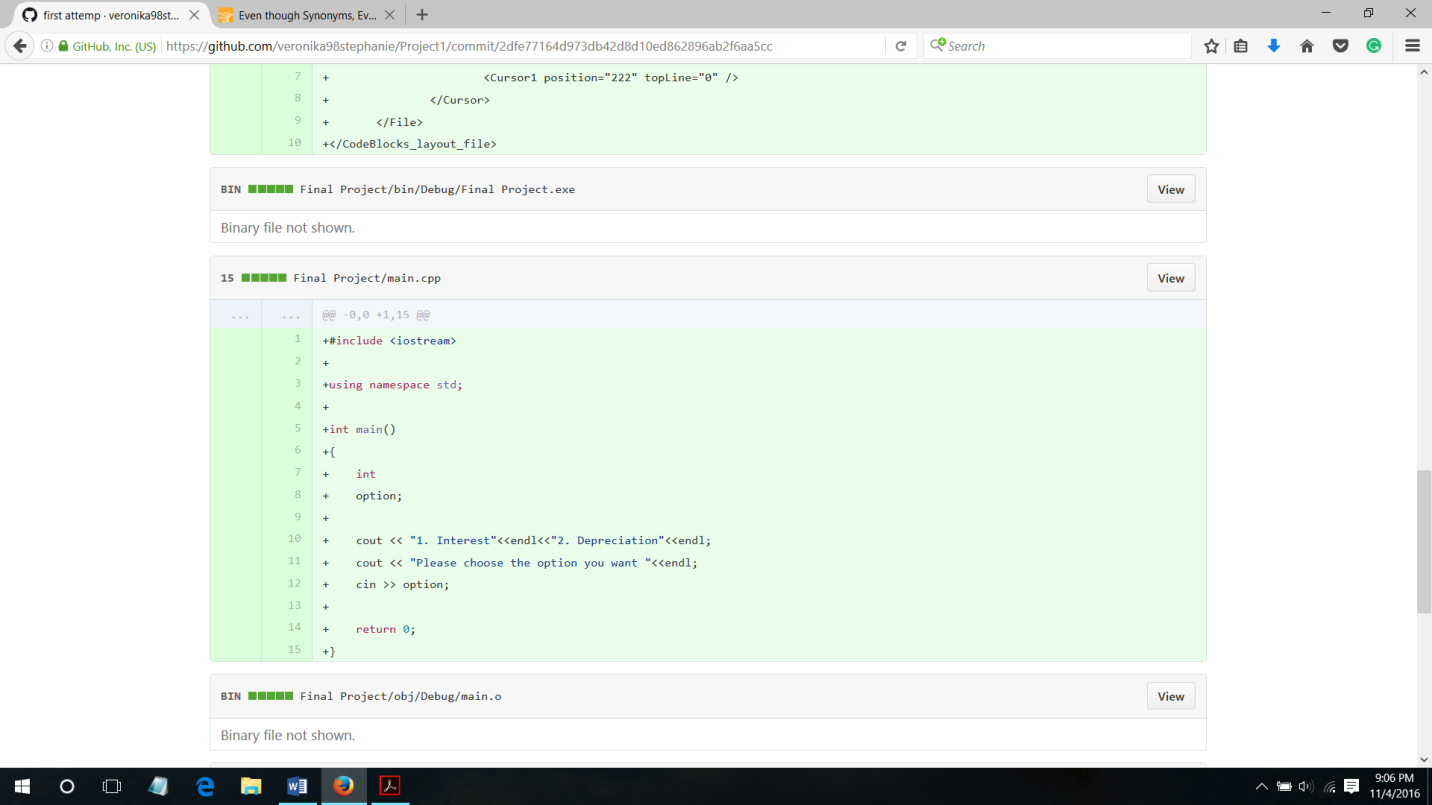
|  |
| --- |
| Interest |
| **-**compoundInt |
| +Interest(p:double, c:double, m:double) |
| +getSingleInterest ()const:double |
| +Interest (double, double, double, int); |

|  |
| --- |
| Depreciation |
| +Depreciation( p:double, c:double, m:double ) |
| +getDepreciationGoodValue ()const: double |
| +getDepreciationBookValue ()const: double |

**III.a. Lessons that Have Been Learned**

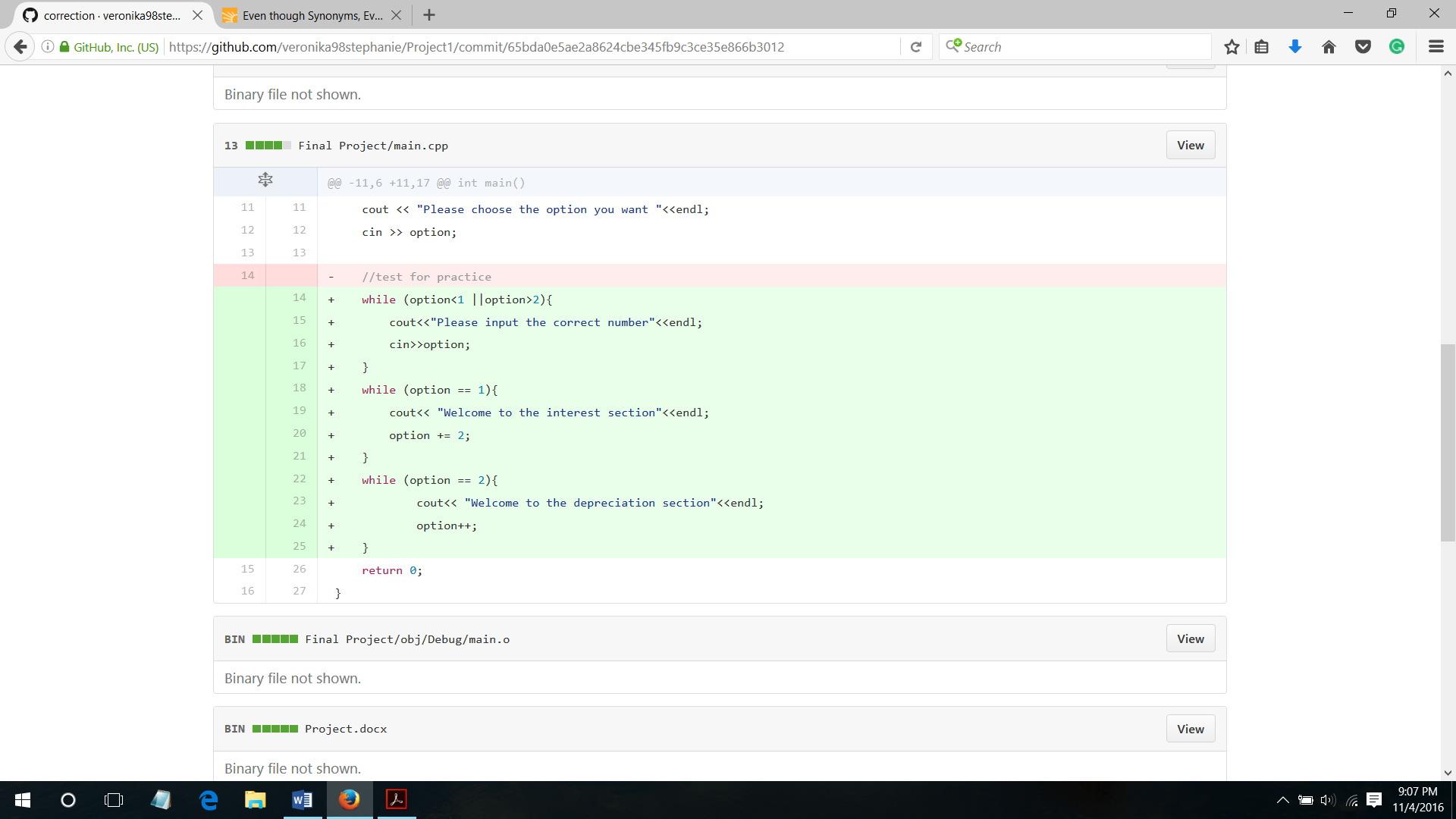
6th of October,

The first time I uploaded my final project progress using Git Bash. The source code was still simple, only using cout for output and cin for user’s input.

 Although it was very simple, there is still an important lesson to be learnt. It is a practice to use the right symbol for cin and cout. Also, to learn the basic things that are needed for a program in order to run, for example include library, using namespace, cout, and int main in order to print the hello world on the screen.

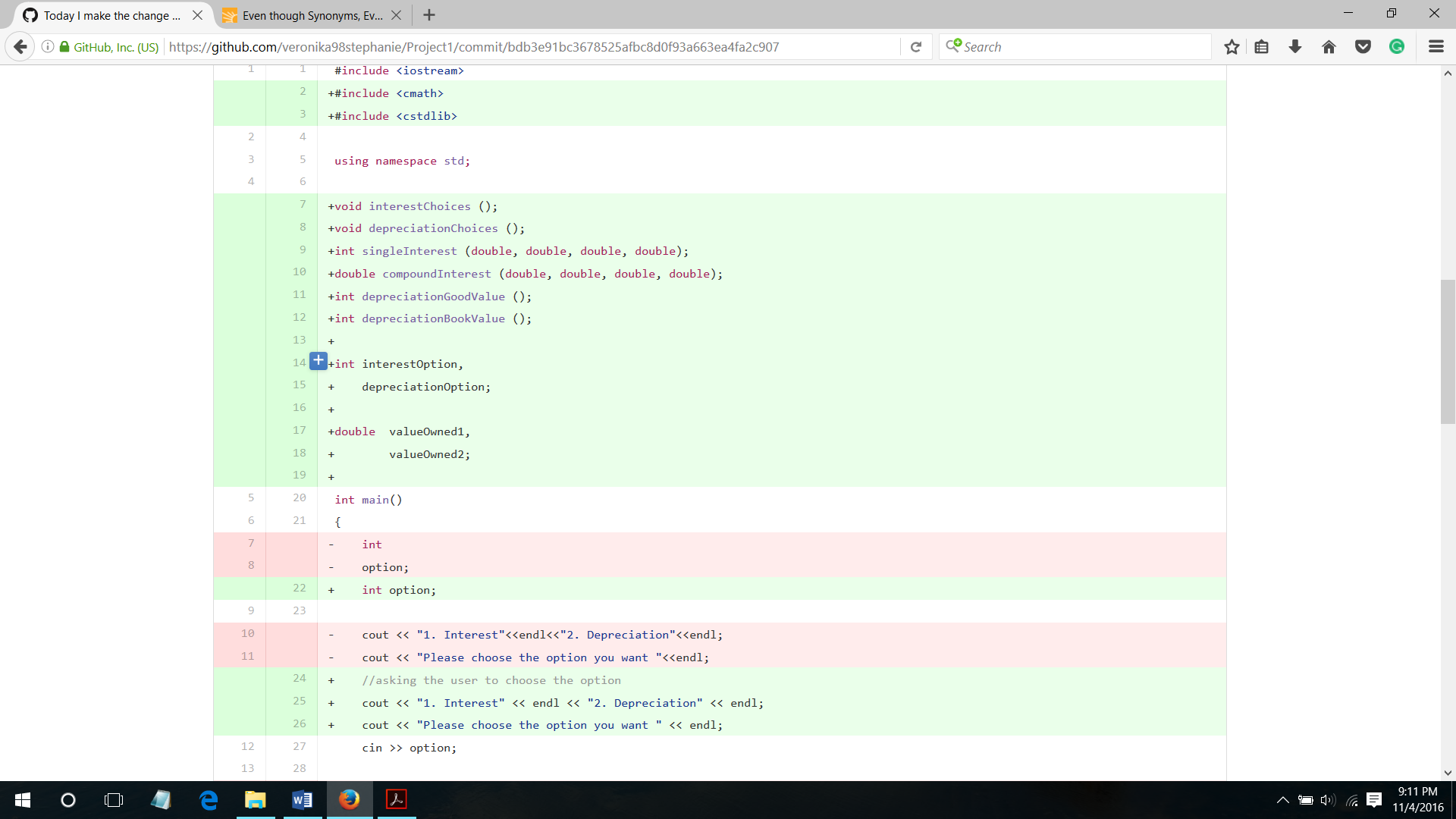
10th of October,

Added several things in the project such as the while loop for the input validation and option. I just realized that using while instead of if for the option is not right. Although the program could perform well. Things that I have learned is to use the right method. For instance, using if instead of while for the option so it would not iterate over and over again if the option +2 and option++ was not declared.

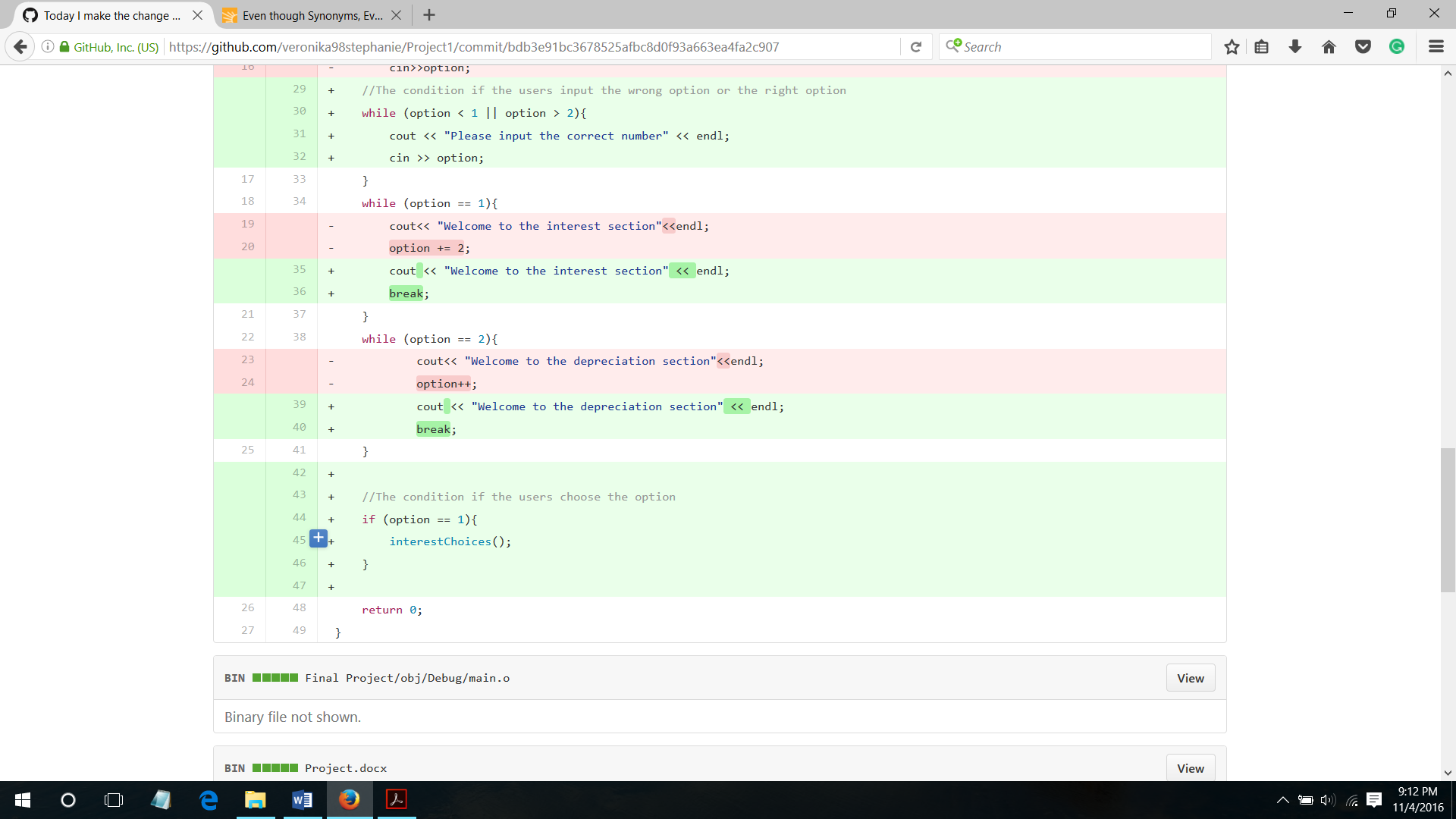


13th of October,

I declared several functions, and using if for the further option. Despite of declaring the functions, I had not defined the functions, but at least I had learnt how to declare the functions before the int main.

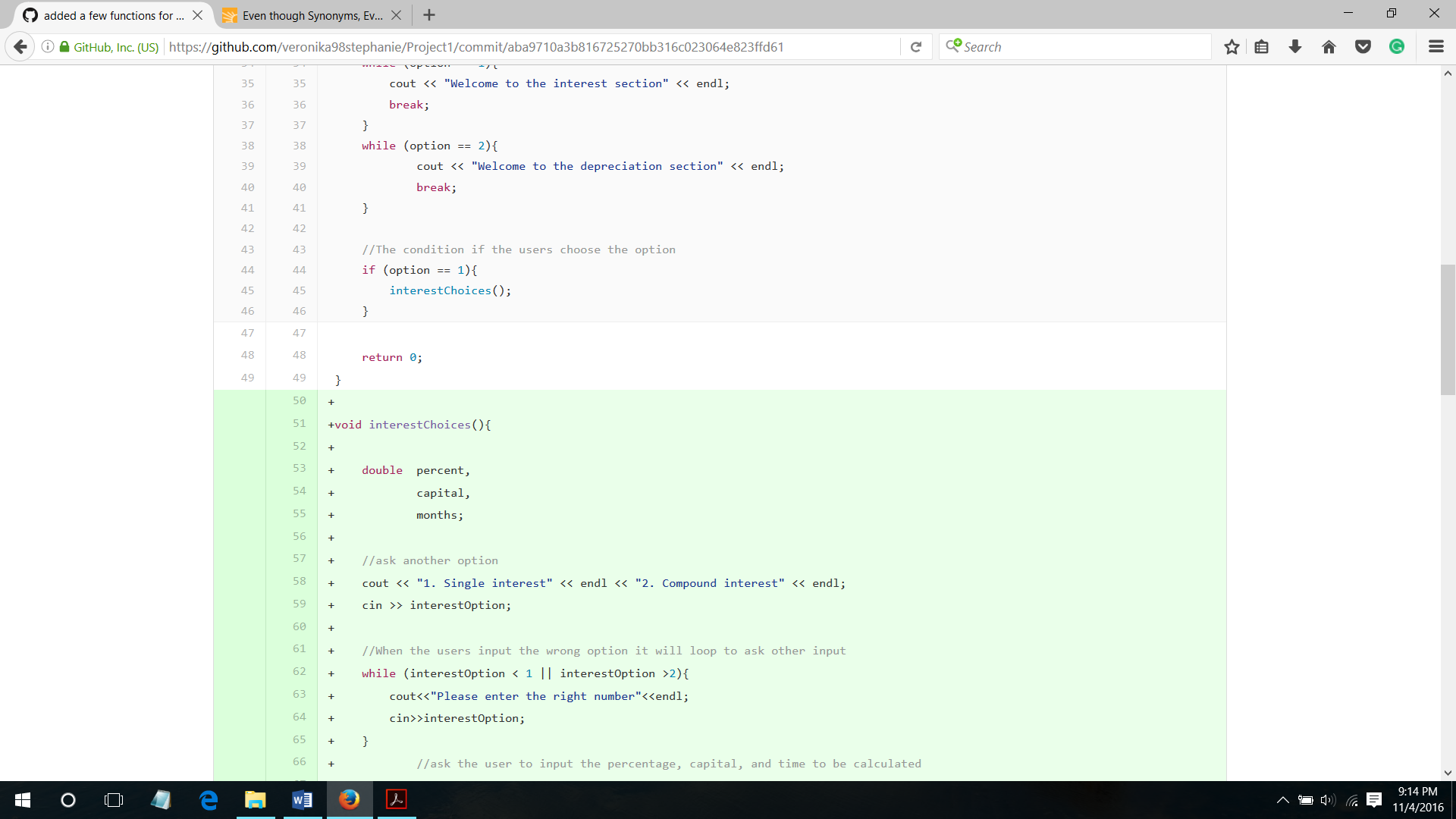


I also learned about how to use break for while loop and learned the difference between “=” for assigning value and “==” to compare values.



18th of October,

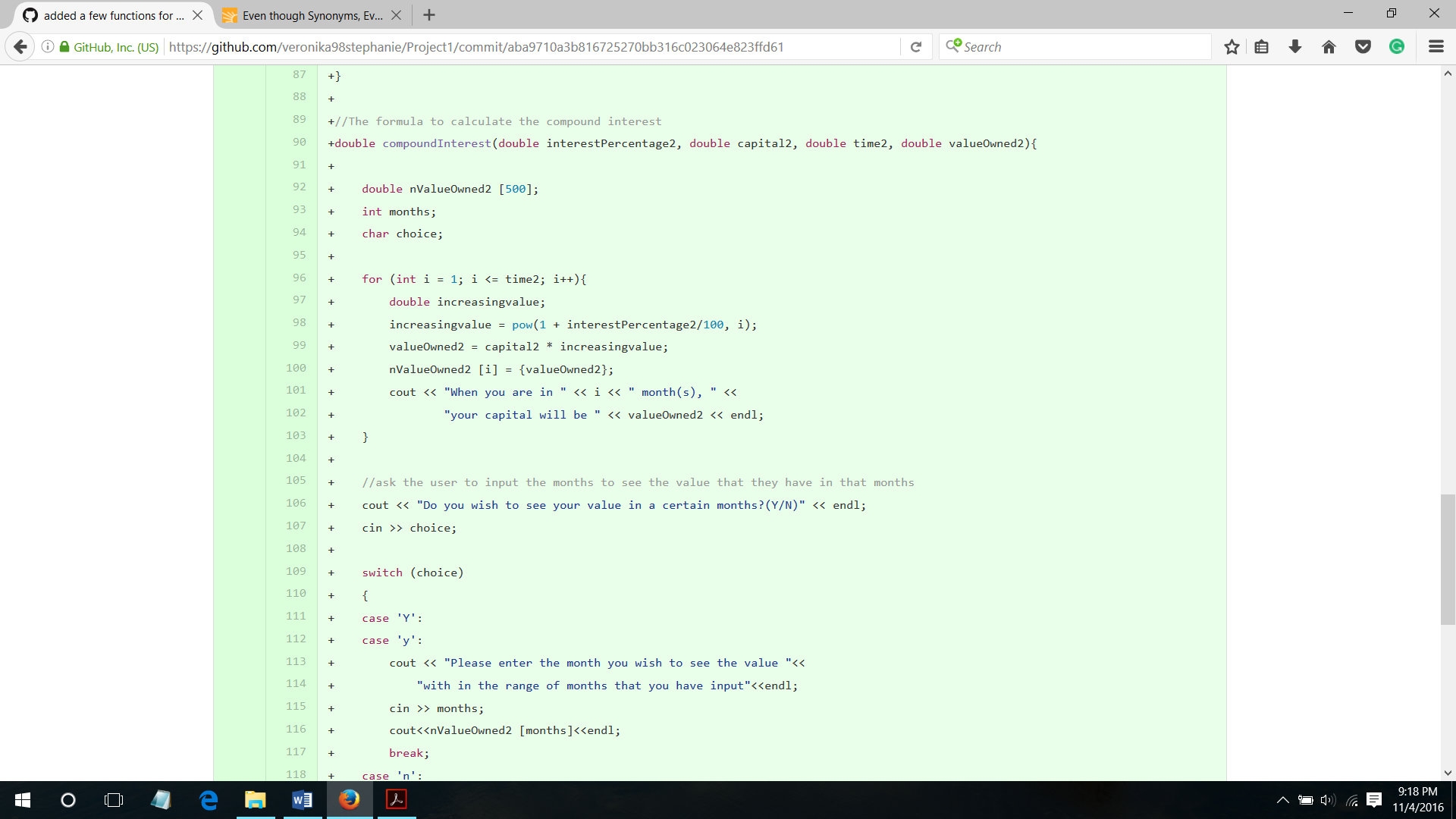
After the declaration of several functions on 13th of October, I defined the interest function under the int main function and expand the options from interest to single and compound interest.



I also expand my knowledge about other ways to check the option. I learned to use switch instead of if , else if , and else. Besides that, I planned to make the menu run only once, so for the compound interest option when the user do not want to check for their value for a certain month, the program will exit (I used exit(0) and cstdlib library).

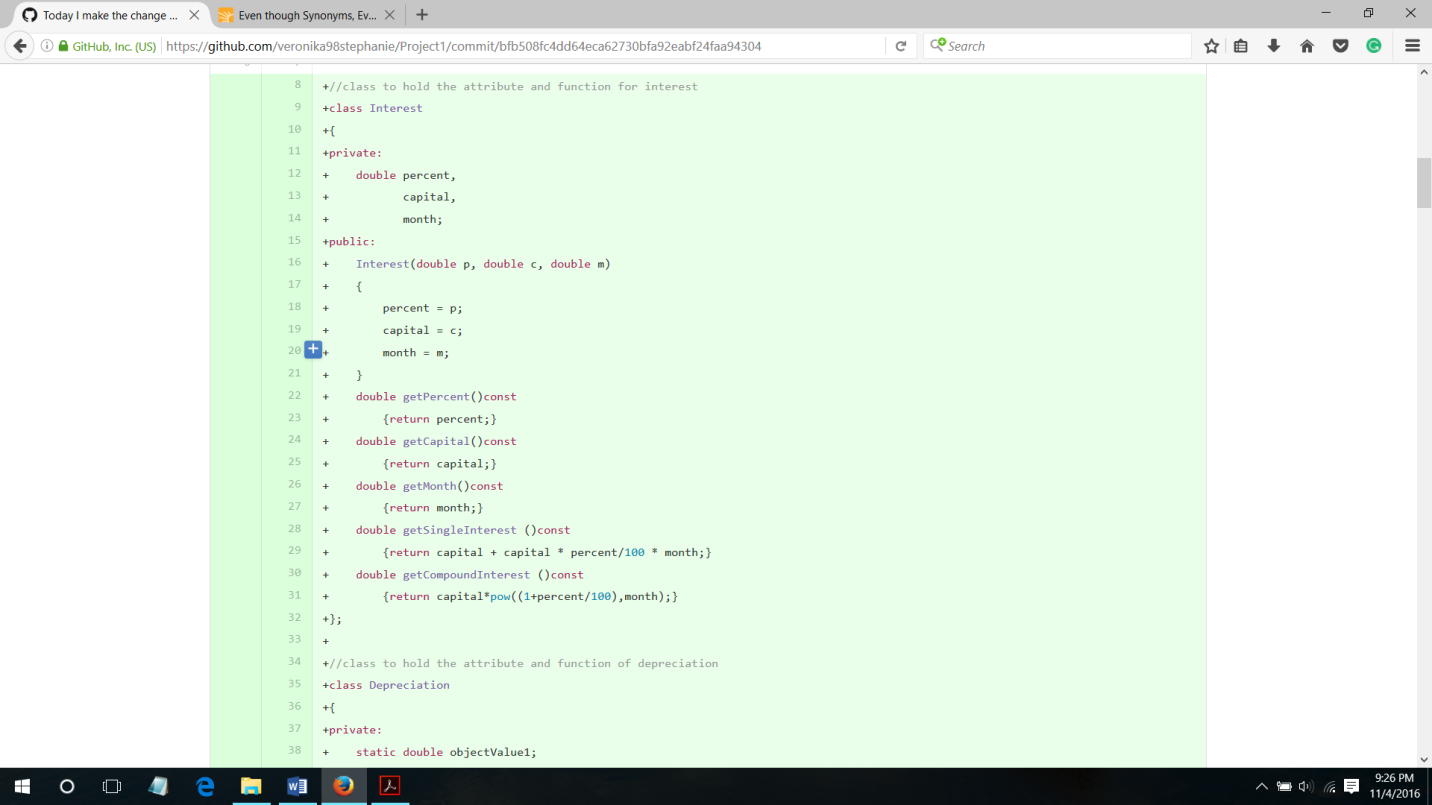
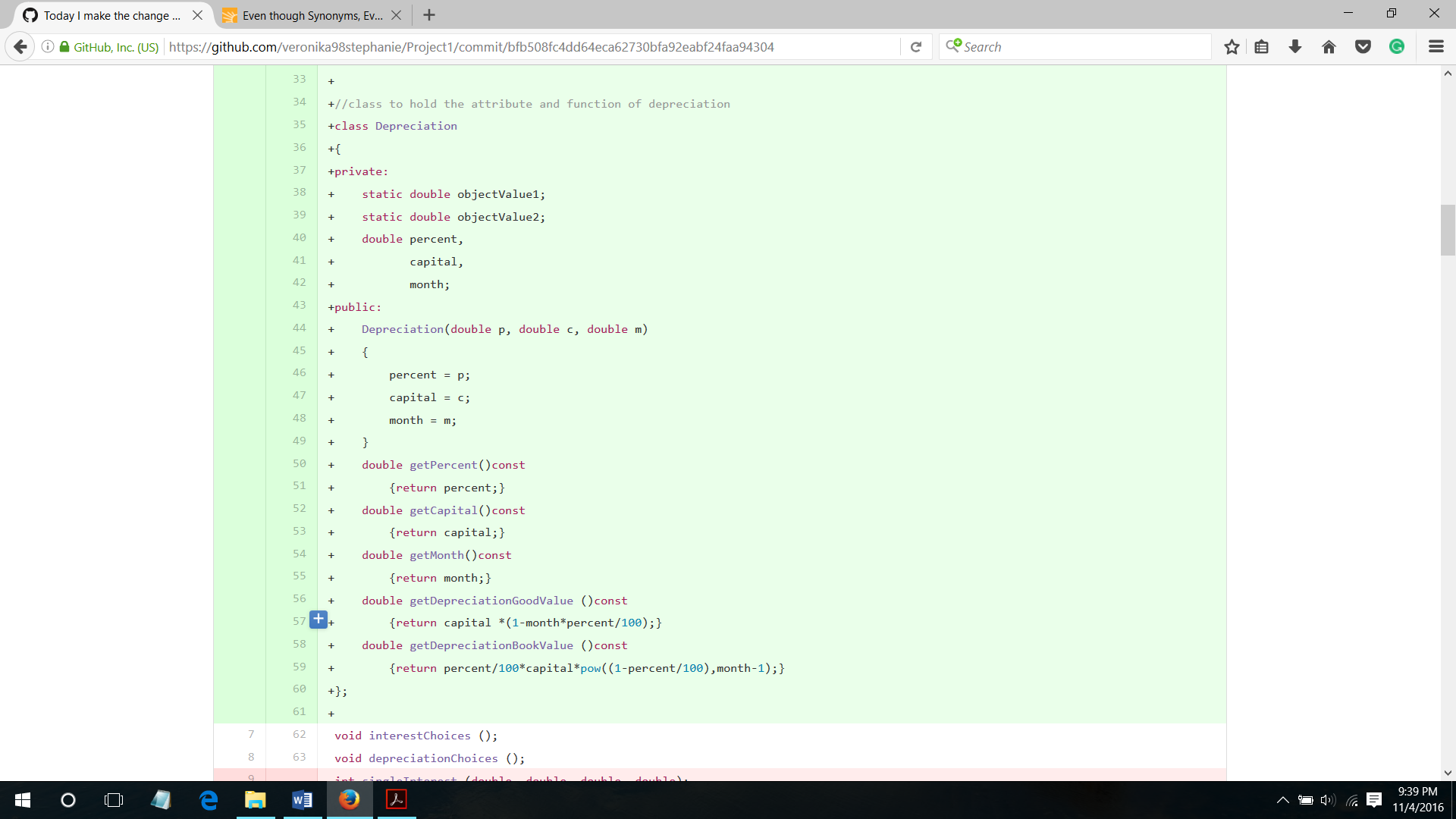


The last thing, to print the value of compound interest for each month I used array.

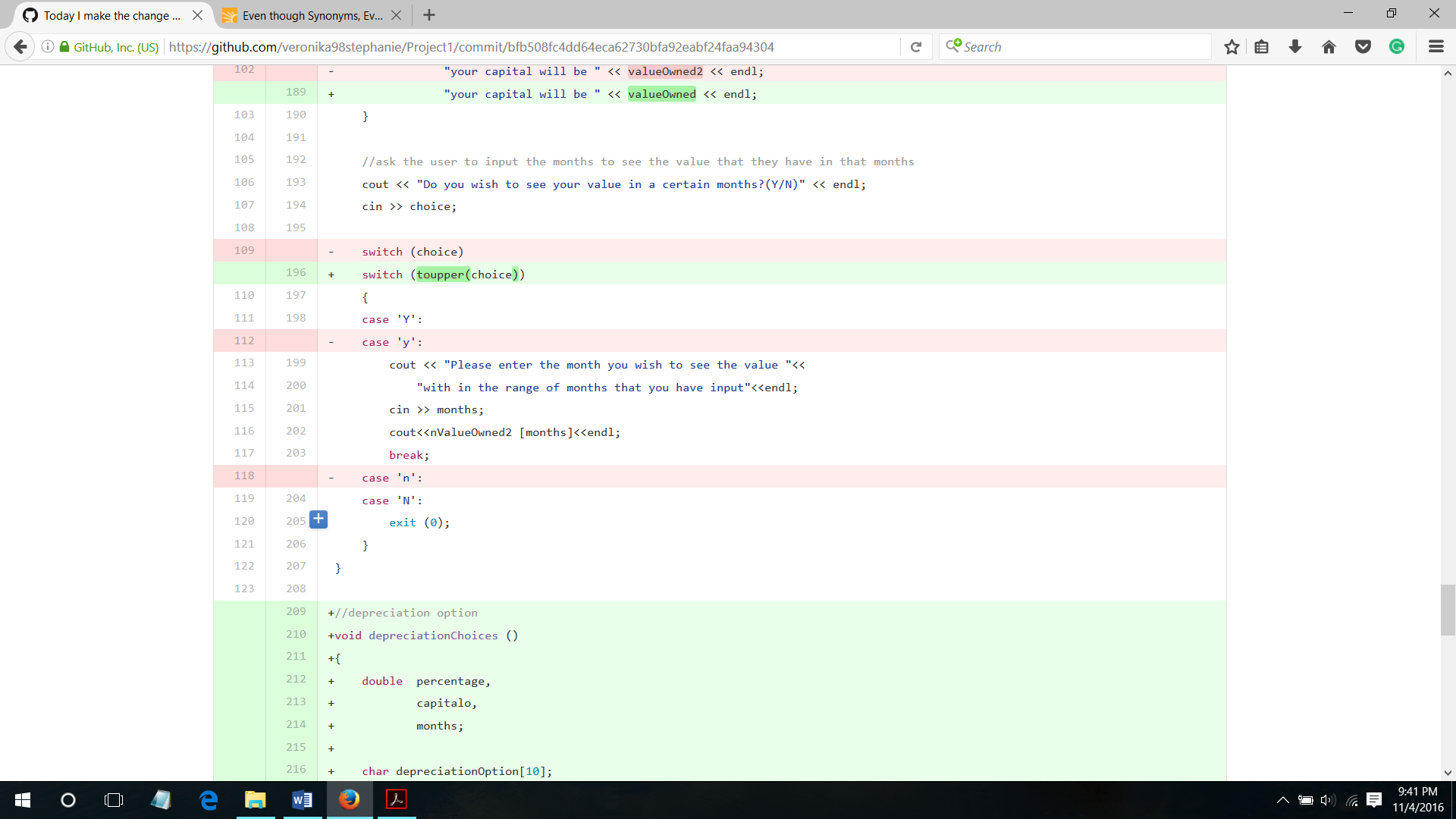
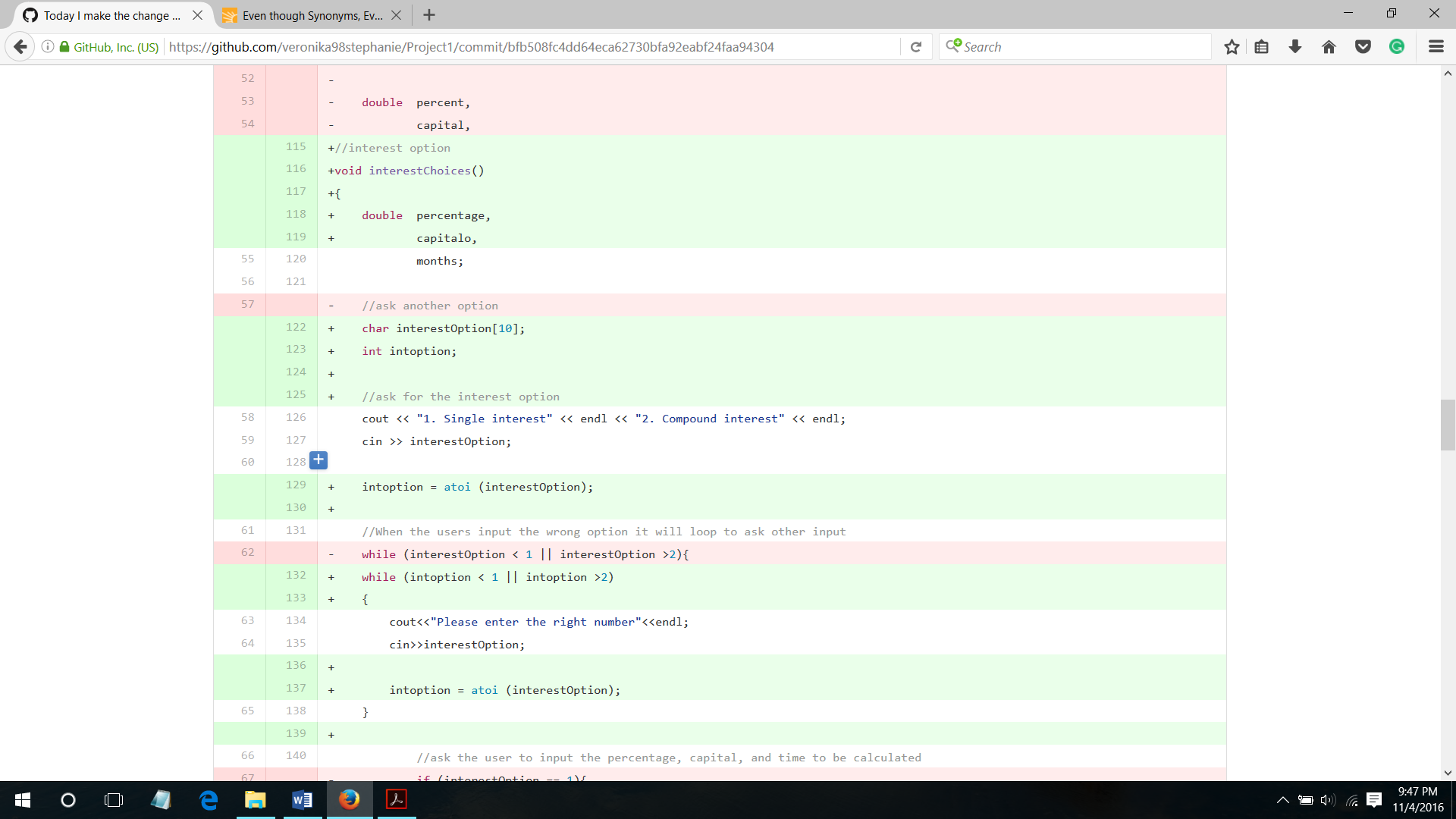


25th of October,

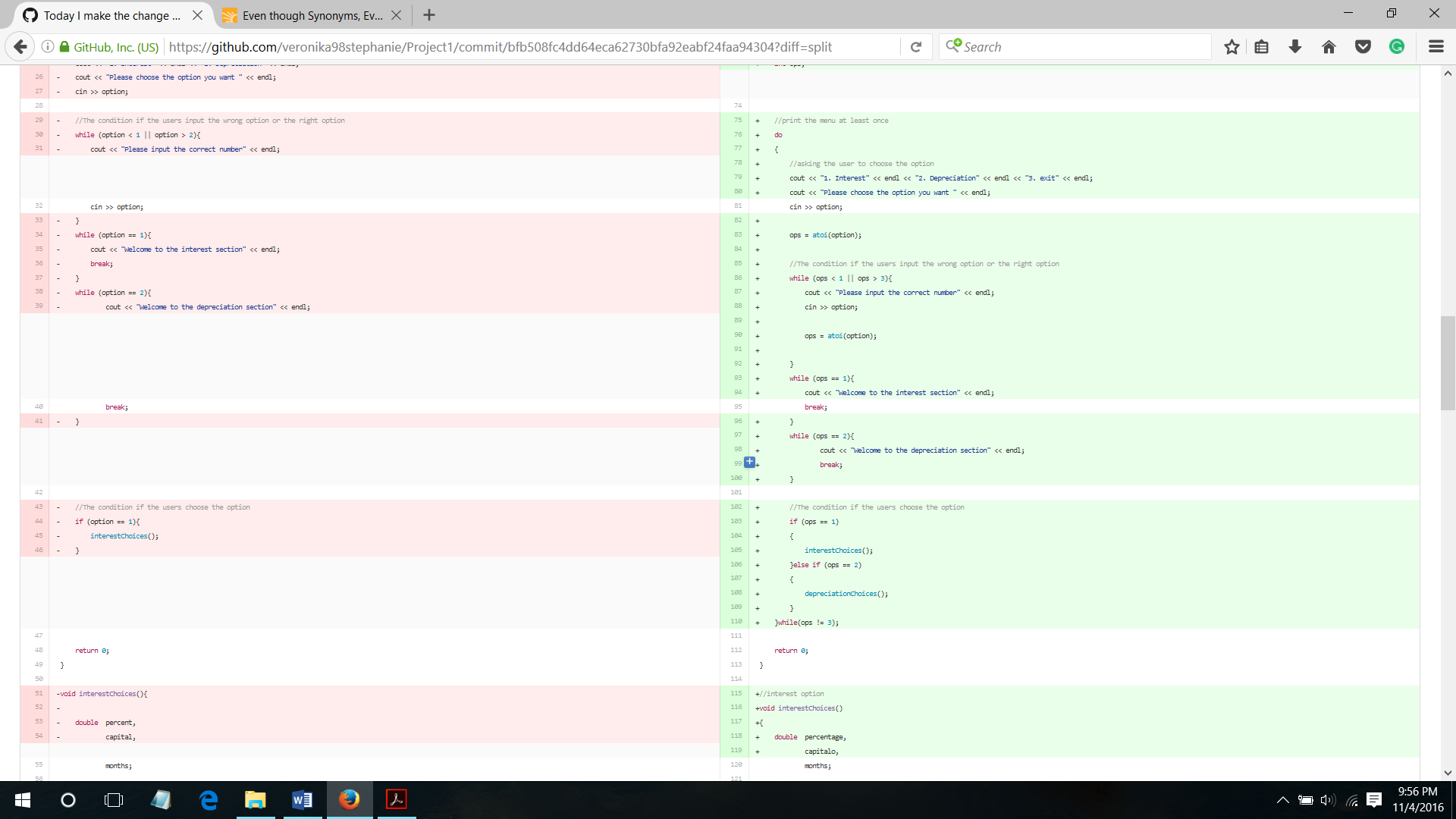
I changed most of my code because of I had learned about class, so I implementing what I had studied. So I made two classes named Interest and depreciation. I declared both of them before the int main and after the using namespace std. Both of them contains setter and getter to access the private member.

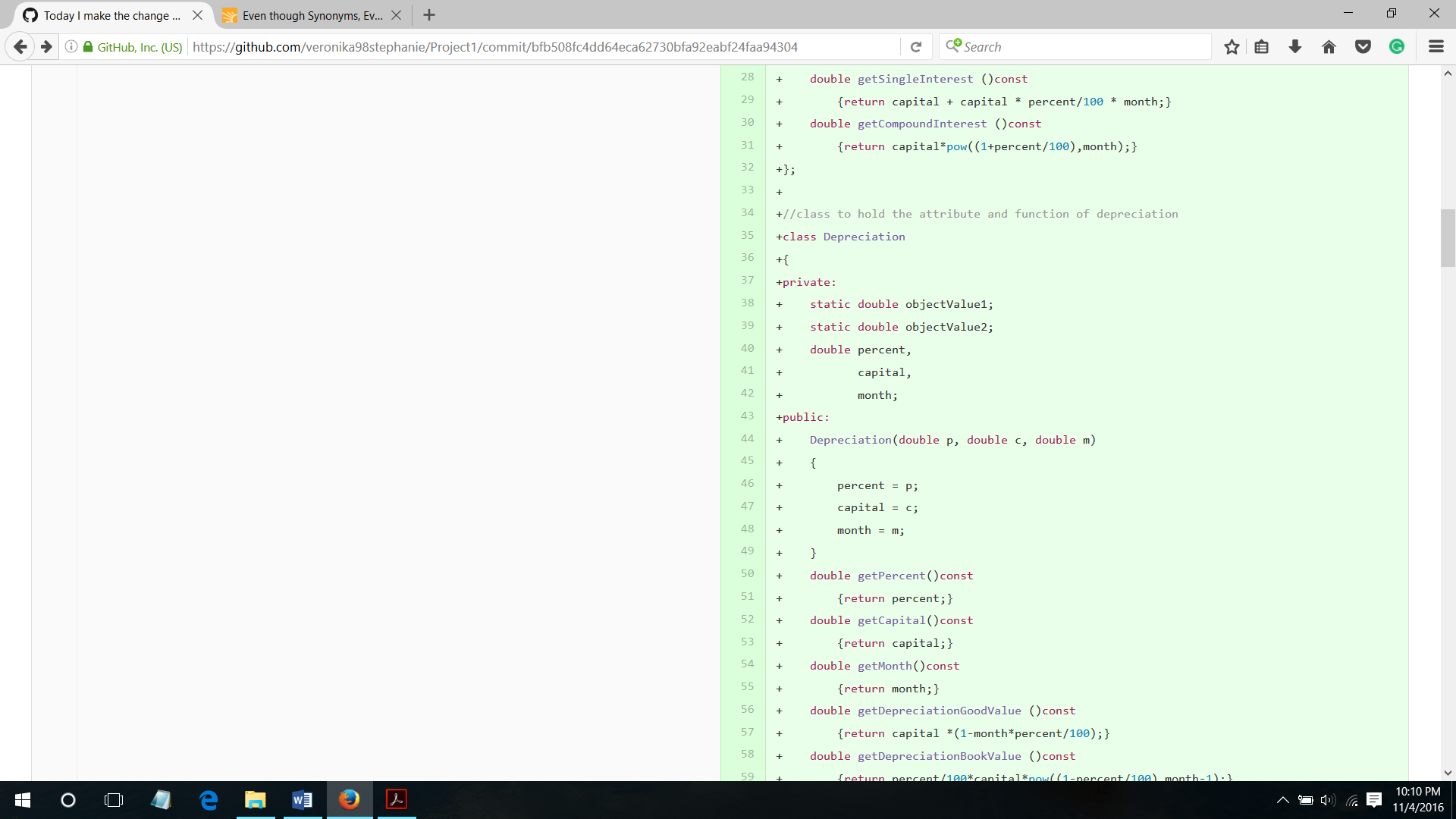
I also learned how to use toupper function to convert the choice into uppercased alphabet so that I would not need to write two condition for each of the alphabet and atoi function to convert the character choice into an integer.



I used do while for the first time in my project in order to show the menu at least once.

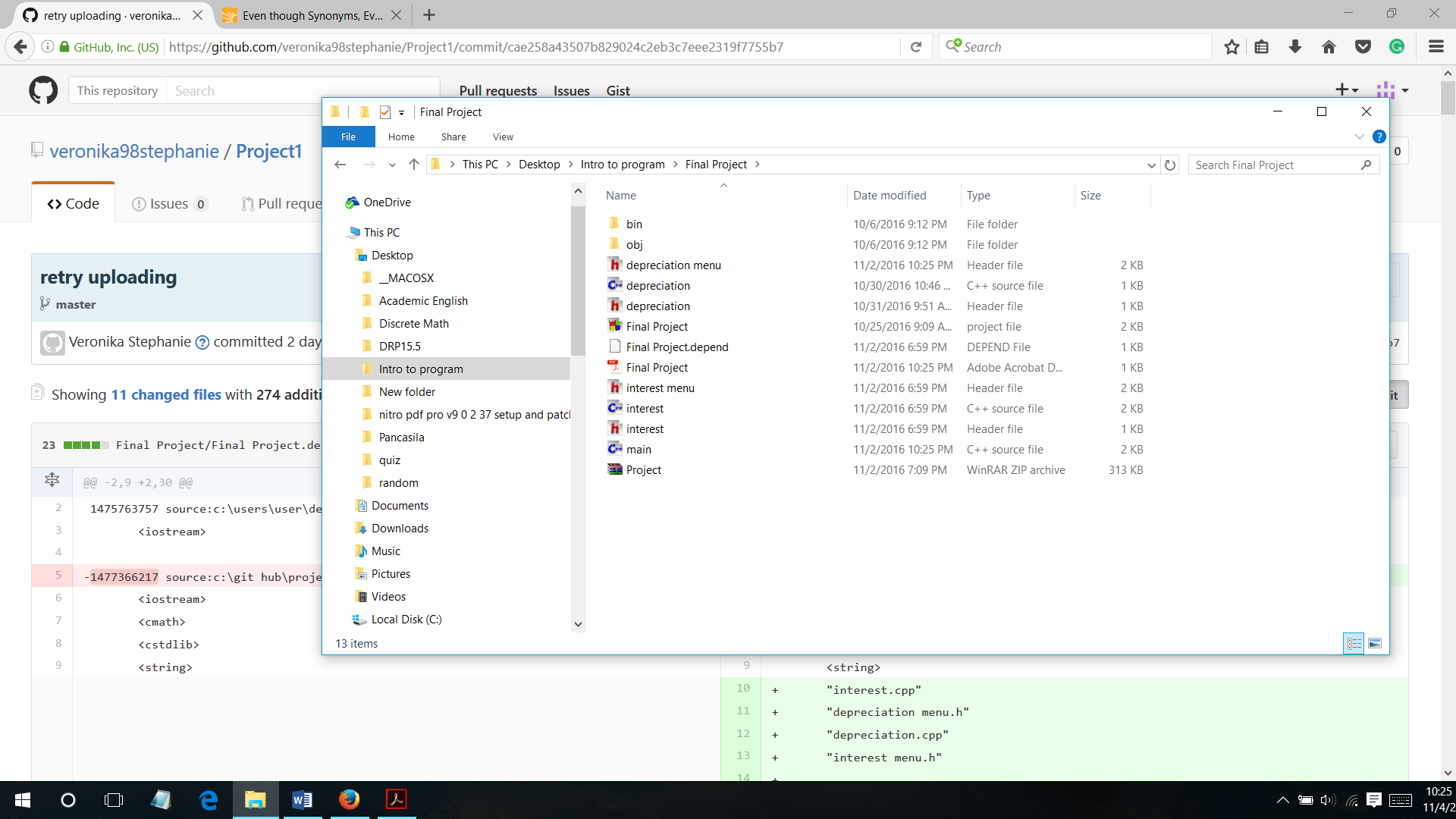


The last thing that I learned was static member even though I only declared it (not using it), I know that the static member does not belongs to any instance of a class, even we do not need to create an instance in order to have a value to be stored in that static member variable.



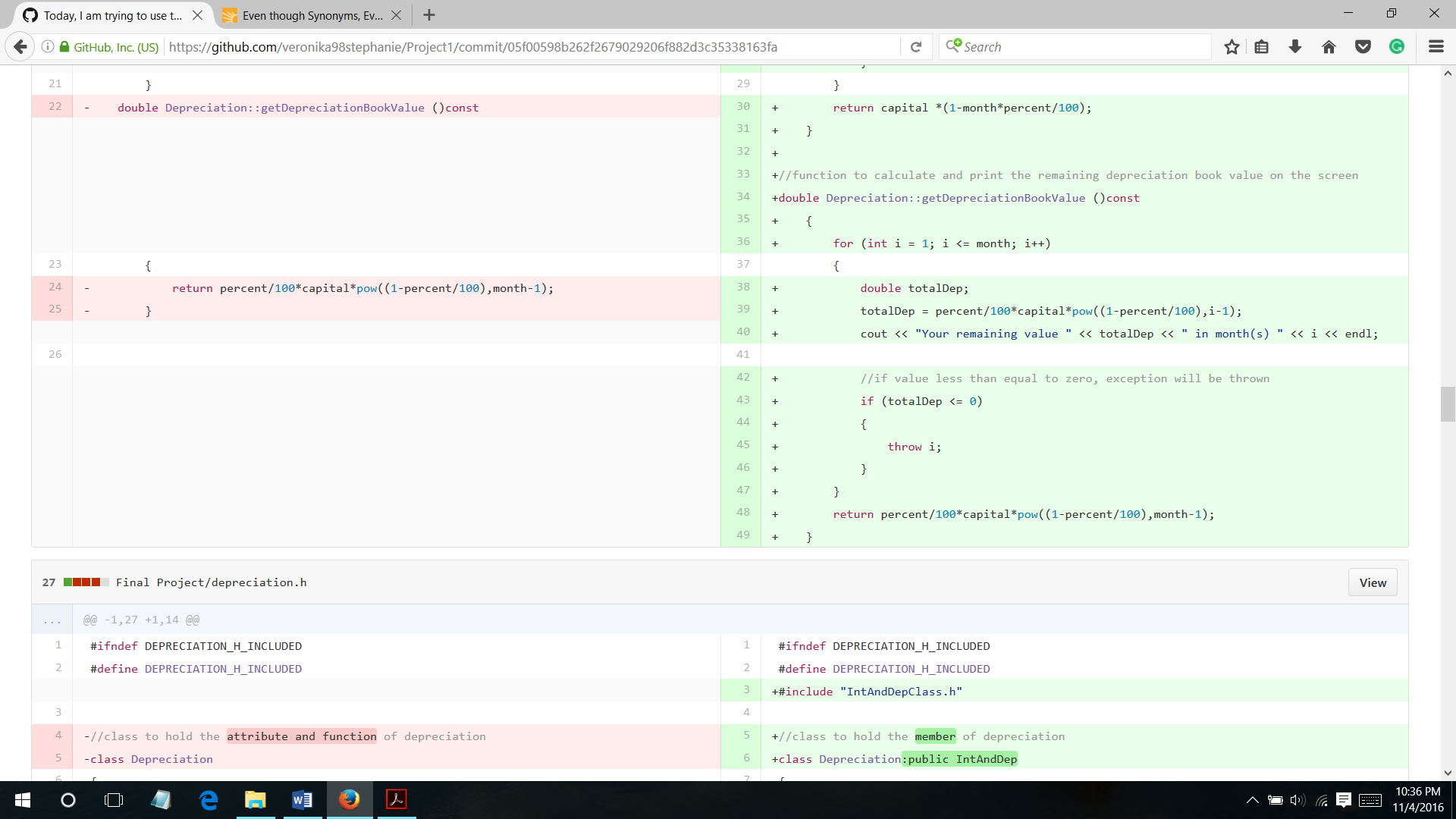
2nd of October,

I started to separate my code into several files, for the class I used cpp and header file. Header for the class specification (class and class members declaration), and cpp for the class implementation ( class member function definition).



4th of October,

I tried to implemented the method that I had learned on 2nd of October (try and catch).



**III.b. Problem that Have Been Overcome**

Making this program is not easy for me. First of all, when I was asked to made a project, I did not have any basic in programming. So, it was a little bit confusing on choosing the right project for myself. On the other hand, we were given a deadline to upload our progress every week. Luckily, I was capable to upload my progress on time. There were many other problems that I had in the process of making this project, the worst cases were the logical error. The IDE could not detect the error, unlike the syntax error. It took long time to fix the error. Furthermore, developing a simple program that had pop into our mind for the first time to a more advance project was not easy. For instance, in my case, I changed almost whole of my project when I wanted to change interest and depreciation function into class. After that, when I tried to implement the new material that I had learnt usually failure comes first. Then, I had to try and test it several times before it run.

**V. Source Code**

1. ***Main.cpp***

#include <iostream>

#include <cmath>

#include <cstdlib>

#include <string>

#include "interest.cpp"

#include "depreciation menu.h"

#include "depreciation.cpp"

#include "interest menu.h"

using namespace std;

void interestChoices ();

void depreciationChoices ();

void exitchoice()

{

char choice;

cout << "Do you wish to Exit? (Y/N)" << endl;

cin >> choice;

if (toupper(choice)=='Y')

{

exit(0);

}

else

{

return;

}

}

int main()

{

char option[10];

int ops;

//print the menu at least once

do

{

system("cls");

//asking the user to choose the option

cout << "1. Interest" << endl << "2. Depreciation" << endl << "3. exit" << endl;

cout << "Please choose the option you want " << endl;

cin >> option;

ops = atoi(option);

//The condition if the users input the wrong option or the right option

while (ops < 1 || ops > 3)

{

cout << "Please input the correct number" << endl;

cin >> option;

ops = atoi(option);

}

if (ops == 1){

cout << "Welcome to the interest section" << endl;

}

if (ops == 2){

cout << "Welcome to the depreciation section" << endl;

}

//The condition if the users choose the option

if (ops == 1)

{

interestChoices();

}

else if (ops == 2)

{

depreciationChoices();

}

}while(ops != 3);

return 0;

}

1. ***IntAndDepClass.h***

#ifndef INTANDDEPCLASS\_H\_INCLUDED

#define INTANDDEPCLASS\_H\_INCLUDED

class IntAndDep

{

protected:

double percent,

capital,

month;

public:

double getPercent()const

{

return percent;

}

double getCapital()const

{

return capital;

}

double getMonth()const

{

return month;

}

};

#endif // INTANDDEPCLASS\_H\_INCLUDED

1. ***Interest.h***

#ifndef INTEREST\_H\_INCLUDED

#define INTEREST\_H\_INCLUDED

#include "IntAndDepClass.h"

//class to hold members for interest

class Interest:public IntAndDep

{

private:

double compoundInt;

public:

Interest(double p, double c, double m);

double getSingleInterest ()const;

Interest (double, double, double, int);

};

#endif // INTEREST\_H\_INCLUDED

1. ***Interest.cpp***

#include <iostream>

#include "interest.h"

#include "IntAndDepClass.h"

using namespace std;

Interest::Interest(double p, double c, double m)

{

this->percent = p;

this->capital = c;

this->month = m;

}

double Interest::getSingleInterest ()const

{

double totalInt;

for (int i = 1; i <= month; i++)

{

totalInt = capital + capital \* percent/100 \* i;

cout << "In month " << i << ", your capital will be " << totalInt << endl;

}

return totalInt;

}

Interest::Interest(double p, double c, double m, int a )

{

this->percent = p;

this->capital = c;

this->month = m;

double \*ptr = &compoundInt;

ptr = new double[a];

for (int i = 0; i < a ; i++)

{

\*(ptr + i) = c\*pow((1+p/100), i+1);

cout << "In month " << i+1 << ", your capital will be " << ptr[i] << endl;

}

delete []ptr;

}

1. ***Interest menu.h***

#ifndef INTEREST\_MENU\_H\_INCLUDED

#define INTEREST\_MENU\_H\_INCLUDED

#include <iomanip>

void exitchoice();//defined in main.cpp

//interest option

void interestChoices()

{

double percentage,

capitalo,

months;

char interestOption[10];

int intoption;

//ask for the interest option

cout << "1. Single interest" << endl << "2. Compound interest" << endl << "3. Back" << endl;

cin >> interestOption;

intoption = atoi (interestOption);

//When the users input the wrong option it will loop to ask other input

while (intoption < 1 || intoption >3)

{

cout<<"Please enter the right number"<<endl;

cin>>interestOption;

intoption = atoi (interestOption);

}

//ask the user to input the percentage, capital, and time to be calculated

if (intoption == 1)

{

cout << "-SINGLE INTEREST-" << endl;

cout << "Percentage....%per month(s),";

cout << " the capital...., and how long....month(s)" << endl;

cin >> percentage >> capitalo >> months;

Interest single(percentage, capitalo, months);

system("cls");

cout << "Your capital will be " << single.getSingleInterest() << endl;

cout << "-USER INPUT DATA-"<< endl;

cout << left << setw (10) << "Month" << right << setw(10) << ": " << single.getMonth() << endl;

cout << left << setw (10) << "Capital" << right << setw(10) << ": " << single.getCapital() << endl;

cout << left << setw (10) << "Percentage" << right << setw(10) << ": " << single.getPercent() << "%" << endl;

exitchoice();

}

else if (intoption == 2)

{

cout << "-COMPOUND INTEREST-" << endl;

cout << "Please input the interest percentage....%per month(s),";

cout << " the capital...., and how long....month(s)" << endl;

cin >> percentage >> capitalo >> months;

system("cls");

Interest compound(percentage, capitalo, months, months);

cout << "-USER INPUT DATA-"<< endl;

cout << left << setw (10) << "Month" << right << setw(10) << ": " << compound.getMonth() << endl;

cout << left << setw (10) << "Capital" << right << setw(10) << ": " << compound.getCapital() << endl;

cout << left << setw (10) << "Percentage" << right << setw(10) << ": " << compound.getPercent() << "%" << endl;

exitchoice();

}

else

return;

}

#endif // INTEREST\_MENU\_H\_INCLUDED

1. ***Depreciation.h***

#ifndef DEPRECIATION\_H\_INCLUDED

#define DEPRECIATION\_H\_INCLUDED

#include "IntAndDepClass.h"

//class to hold the member of depreciation

class Depreciation:public IntAndDep

{

public:

Depreciation(double p, double c, double m);

double getDepreciationGoodValue ()const;

double getDepreciationBookValue ()const;

};

#endif // DEPRECIATION\_H\_INCLUDED

1. ***Depreciation.cpp***

#include <iostream>

#include "depreciation.h"

#include "IntAndDepClass.h"

using namespace std;

Depreciation::Depreciation(double p, double c, double m)

{

percent = p;

capital = c;

month = m;

}

//function to calculate and print the remaining depreciation good value on the screen

double Depreciation::getDepreciationGoodValue ()const

{

//check whether the value less than zero;

for (int i = 1; i <= month; i++)

{

double totalDep;

totalDep = capital \*(1-i\*percent/100);

cout << "Your remaining value " << totalDep << " in month(s) " << i << endl;

//if value less than equal to zero, exception will be thrown

if (totalDep <= 0)

{

throw i;

}

}

return capital \*(1-month\*percent/100);

}

//function to calculate and print the remaining depreciation book value on the screen

double Depreciation::getDepreciationBookValue ()const

{

for (int i = 1; i <= month; i++)

{

double totalDep;

totalDep = percent/100\*capital\*pow((1-percent/100),i-1);

cout << "Your remaining value " << totalDep << " in month(s) " << i << endl;

//if value less than equal to zero, exception will be thrown

if (totalDep <= 0)

{

throw i;

}

}

return percent/100\*capital\*pow((1-percent/100),month-1);

}

1. ***Depreciation menu.h***

#ifndef DEPRECIATION\_MENU\_H\_INCLUDED

#define DEPRECIATION\_MENU\_H\_INCLUDED

#include "depreciation.h"

#include <iomanip>

void exitchoice();//defined in main.cpp

//depreciation option

void depreciationChoices ()

{

double percentage,

capitalo,

months;

bool wrongInput = true;

char depreciationOption[10];

int depoption;

//ask for the depreciation option

cout << "1. Depreciation good value" << endl << "2. Depreciation book value" << endl << "3. Back" << endl;

//char input: to prevent error if input other than number

cin >> depreciationOption;

depoption = atoi (depreciationOption);

//When the users input the wrong option it will loop to ask other input

while (depoption < 1 || depoption >3)

{

cout<<"Please enter the right number"<<endl;

cin>>depreciationOption;

depoption = atoi(depreciationOption);

}

//ask the user to input the percentage, capital, and time to be calculated

if (depoption == 1)

{

cout << "-DEPRECIATION GOOD VALUE-" << endl;

cout << "Input the percentage....%per month(s),";

cout << " the capital...., and how long....month(s)" << endl;

while (wrongInput)

{

cin >> percentage >> capitalo >> months;

if (percentage<=0||capitalo<=0||months<=0)

{

cout << "Do not enter negative value, please try again" << endl;

wrongInput = true;

}

else

wrongInput = false;

}

Depreciation good(percentage, capitalo, months);

system("cls");

try

{

good.getDepreciationGoodValue();

}

catch (int i)

{

cout << "Your thing will not have any value from month " << i << endl;

}

cout << "-USER INPUT DATA-"<< endl;

cout << left << setw (10) << "Month" << right << setw(10) << ": " << good.getMonth() << endl;

cout << left << setw (10) << "Capital" << right << setw(10) << ": " << good.getCapital() << endl;

cout << left << setw (10) << "Percentage" << right << setw(10) << ": " << good.getPercent() << "%" << endl;

exitchoice();

}

else if (depoption == 2)

{

cout << "-DEPRECIATION BOOK VALUE-" << endl;

cout << "Please input the interest percentage....%per month(s),";

cout << " the capital...., and how long....month(s)" << endl;

while (wrongInput)

{

cin >> percentage >> capitalo >> months;

if (percentage<=0||capitalo<=0||months<=0)

{

cout << "Do not enter negative value, please try again" << endl;

wrongInput = true;

}

else

wrongInput = false;

}

Depreciation book(percentage, capitalo, months);

system("cls");

try

{

book.getDepreciationBookValue();

}

catch (int i)

{

cout << "Your thing will not have any value from month " << i << endl;

}

cout << "-USER INPUT DATA-"<< endl;

cout << left << setw (10) << "Month" << right << setw(10) << ": " << book.getMonth() << endl;

cout << left << setw (10) << "Capital" << right << setw(10) << ": " << book.getCapital() << endl;

cout << left << setw (10) << "Percentage" << right << setw(10) << ": " << book.getPercent() << "%" << endl;

exitchoice();

}

else

return;

}

#endif // DEPRECIATION\_MENU\_H\_INCLUDED