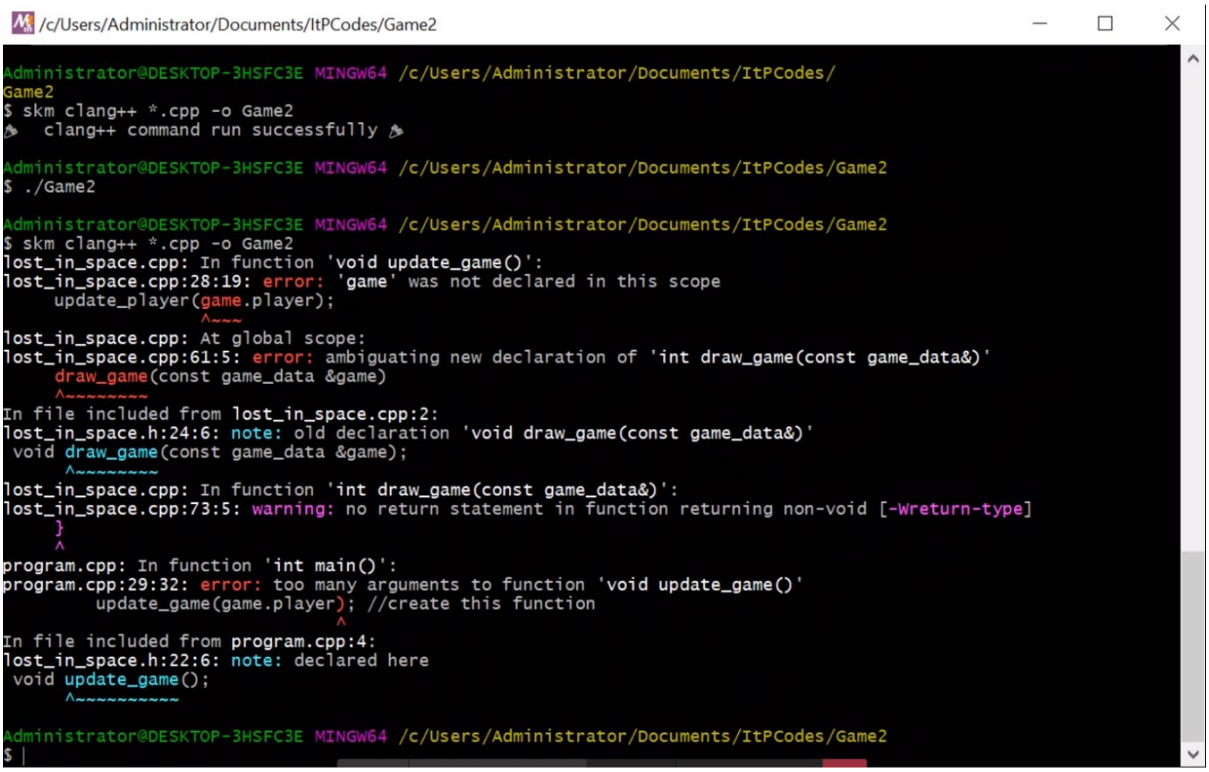
Helping Others



/c/Users/Administrator/Documents/ItPCodes/Game2

```
Administrator@DESKTOP-3HSFC3E MINGW64 /c/Users/Administrator/Documents/ItPCodes/
Game2
$ skm clang++ *.cpp -o Game2
   clang++ command run successfully

Administrator@DESKTOP-3HSFC3E MINGW64 /c/Users/Administrator/Documents/ItPCodes/Game2
$ ./Game2

Administrator@DESKTOP-3HSFC3E MINGW64 /c/Users/Administrator/Documents/ItPCodes/Game2
$ skm clang++ *.cpp -o Game2
lost_in_space.cpp: In function 'void update_game()':
lost_in_space.cpp:28:19: error: 'game' was not declared in this scope
     update_player(game.player);
                   ^~~~
lost_in_space.cpp: At global scope:
lost_in_space.cpp:61:5: error: ambiguating new declaration of 'int draw_game(const game_data&)'
     draw_game(const game_data &game)
     ^~~~~~~~~
In file included from lost_in_space.cpp:2:
lost_in_space.h:24:6: note: old declaration 'void draw_game(const game_data&)'
 void draw_game(const game_data &game);
      ^~~~~~~~~
lost_in_space.cpp: In function 'int draw_game(const game_data&)':
lost_in_space.cpp:73:5: warning: no return statement in function returning non-void [-Wreturn-type]
     }
     ^
program.cpp: In function 'int main()':
program.cpp:29:32: error: too many arguments to function 'void update_game()'
         update_game(game.player); //create this function
                                ^
In file included from program.cpp:4:
lost_in_space.h:22:6: note: declared here
 void update_game();
      ^~~~~~~~~~~

Administrator@DESKTOP-3HSFC3E MINGW64 /c/Users/Administrator/Documents/ItPCodes/Game2
$ |
```

Sarah Masih

Ben's Statement:

Sarah was having several problems with the second part of the game task from week 7.

1. I first discovered that she had tried to include CPP files in some of the headers. We fixed this by removing them and I also explained to her why it is unnecessary to do this and I did this by explaining to her that headers are used to declare artefacts such as functions, structures and enumerations and that CPP files are meant to hold the actual instructions when talking about functions and procedures.
2. Visual Studio Code's intellisense could not identify the game_data struct as well as some data values that were part of the player_data struct correctly. I thought this had something to do with headers being inter-dependent on one another but after fixing that problem, this did not help the intellisense. This however did not prevent the compiler from compiling the game successfully, so we did not worry too much about this problem.
3. Sarah had also incorrectly defined the player variable in her program.cpp file. Initially, she tried to declare it as a game_data struct variable and named it player_data, which is also refers to the player_data struct data type. I simply got her to keep the data type as game_data and got her to rename the variable. I then explained to her how the game_data variable was supposed to work by saying that it is meant to store all of the data about the game being played such as the the player's stats stored in player_data and the power ups array that keeps track of the power ups that are currently spread out throughout the game at any one time.
4. Throughout the code, the header files were not managed correctly. There were header files that were inter-dependent on each other. I explained to Sarah what this meant and how it affects Visual Studio Code and the compiler when this happens. We fixed the problem by making it so

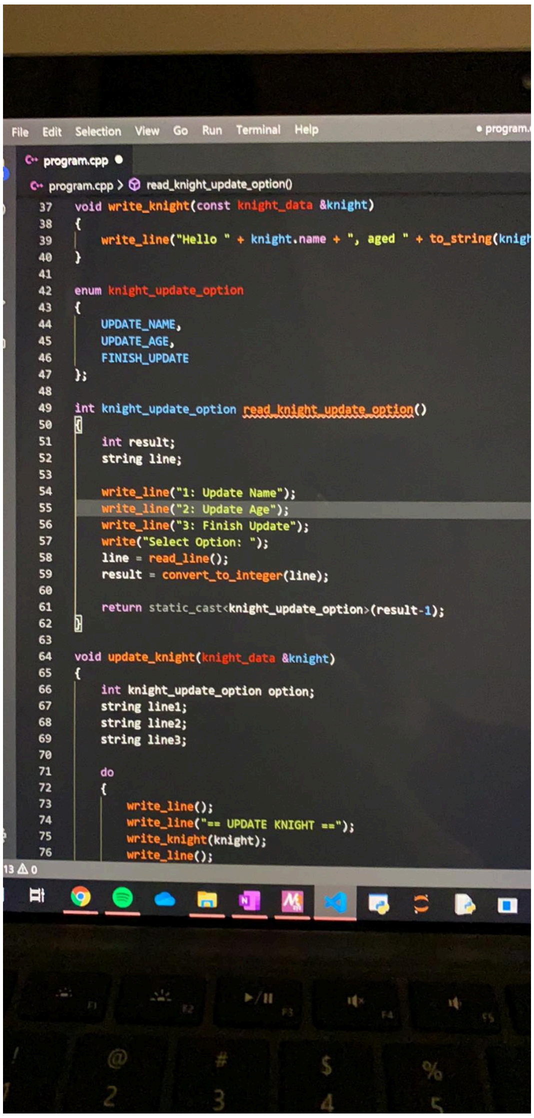
that header files included the splashkit header and the lost in space header included all the headers.

5. I also noticed that between some of the CPP files and their corresponding header files, some functions were missing from their headers which caused compilation errors regarding undefined functions. We fixed this by adding those missing functions into their header files.
6. One final compilation issue we had after fixing all the problems listed above was that there existed to instances of the main() function. Sarah had accidentally left a second copy of program.cpp in her code files for her own reference which ended up confusing the compiler. She fixed this by deleting the extra copy of program.cpp.

Sarah's Statement:

I was getting a very simple yet confusing error about a missing semicolon. Turns out I had included CPP files in header files and included every single file in all the header ones. This major mistake was pointed out by Ben which and he explained that if each header file contains every other file, it would confuse the program as the files are trying to extract information from each other but it is only meant to be one way. I also declared many functions and included libraries and classes where they were not necessary and this had caused even more errors to appear in my program. I had also forgotten to include some functions from the CPP files to the header files which became the reason for another wave of errors. I had a very messy code and Ben helped me clean it out and structure it properly. He was able to find out the reasons for these errors quickly and was able to explain the reason in a clear way.

Signed: **Signed:**


File Edit Selection View Go Run Terminal Help
● program.
program.cpp ●
program.cpp > read_knight_update_option()
13 ⚠ 0

```
void write_knight(const knight_data &knight)
{
    write_line("Hello " + knight.name + ", aged " + to_string(knigh
}

enum knight_update_option
{
    UPDATE_NAME,
    UPDATE_AGE,
    FINISH_UPDATE
};

int knight_update_option read_knight_update_option()
{
    int result;
    string line;

    write_line("1: Update Name");
    write_line("2: Update Age");
    write_line("3: Finish Update");
    write("Select Option: ");
    line = read_line();
    result = convert_to_integer(line);

    return static_cast<knight_update_option>(result-1);
}

void update_knight(knight_data &knight)
{
    int knight_update_option option;
    string line1;
    string line2;
    string line3;

    do
    {
        write_line();
        write_line("== UPDATE KNIGHT ==");
        write_knight(knight);
        write_line();
```

Lael Newton

Ben's Statement:

Lael had made a simple error in her code for the structs and enums task. She attempted to define read_knight_update_option() to be both an integer and knight_update_option function (knight_update_option being the enumeration declaration above the read_knight_update_option() function). She fixed this by removing 'int' at the beginning and defining it a knight_update_option function that returns the option chosen by user in the menu.

Lael's Statement:

Had a very simple issue but could not seem to figure out what the issue was or the solution. Got into contact with Ben, who then helped me identify where the issue came from and why it was producing the error it did. Found that I identified a function as an integer when it contained information from a struct earlier in the code. Even though it was a simple issue, Ben was able to help find it quickly and helped make sure that everything ran like it should after resolving the issue.

Signed:

Signed: