Computing

A Level

Computing
Exemplar Candidate Work

H447

Unit F454 Computing Project C

October 2015

www.ocr.org.uk

OCR
Oxford Cambridge and RSA

Contents

Introduction

This sample material serves as a general guide. It provides the following benefits to a teacher:

- Gives teachers an appreciation of the variety of work that can be produced for this unit
- Shows how the mark scheme has been applied by a senior assessor
- Provides examples of both good and weak application of different parts of the mark scheme
- Provides real examples of work submitted for F454.

It is important to make the point that the teacher support materials play a secondary role to the Specification itself. The Specification is the document on which assessment is based and specifies what content and skills need to be covered in delivering the course. At all times, therefore, this teacher support should be read in conjunction with the Specification. If clarification on a particular point is sought then that clarification should be found in the Specification itself.

This project is an example which has not been scaled or adjusted during the moderation process; as such the marks awarded have been viewed as being 'within tolerance' when assessed by a moderator

URS/Teacher's Comments

Sample Material – Project 'C'

Centre marks/comments:

Definition, Investigation and Analysis

Definition **Max Mark: 3**

There is a sold and descriptive text detailing the area of investigation and the potential problems of the systems that are in place within the role of Text base games. Early issues are hinted at.

Centre Mark Awarded: 3

Investigation and Analysis **Max Mark: 11**

There is clear interatction via the use of online questionnaire with a range of end users across the school. All of the details are recorded/collected. There is a thorough analysis of what people want developed in this game-styled project. The requirements are detailed and relate back to the questions and research gained. HOwever, there is potential for broader research using past games perhaps as a base line. The hardware and software requirements are a little vaige and lack analysis/justification. The hardware/software requirements need more justification.

Centre Mark Awarded: 8

Total Section Mark : 11

Design

Nature of Solution **Max Mark: 6**

The design specification is functional but is not as detailed to allow independent implementation by a 3rd party. Objectives are clear although lack detail.There are some descriptions of the UI deisgn and the function of buttons etc. The menu flow is shown and makes sense. There is evidence that he has gained feedback from the end user at this point. The "map" and general story line is useful. Variables are present but not complete.

Centre Mark Awarded: 4

Algorithms **Max Mark: 5**

The algorithms are funcitonal and describe the basic actions that a player can take in the game. They cover all main actions. However they are not proven to meet the full design specigication, not have they need tested for functionality.

Centre Mark Awarded: 3

Test Strategy **Max Mark: 6**

There is a trivial test plan included that lacks depth or justification. There are a very limited number of test cases and things like testing combat etc are never checked.

Centre Mark Awarded: 2

Total Section Mark: 9

Software Development and Testing

Software Development **Max Mark: 16**

There is a solution to the majoirty of the deisgn work and this is alpha tested as the solution develops. The testing is narrative based, and is not linked closely to the original test plan, which was limited in nature. The screenshots aid the development of the project and ther eis sufficient annotation to illistrate development for purpose. There are annotations within the code, but these could be devleoped to explain the modular function further.

Centre Mark Awarded: 10

Testing **Max Mark: 14**

The testing covers all button pressing and "option" selection testing. There is evidence that the original test plan has been "re-written" and this test plan is much more robust, covering a range of data and options as needed.
End user tests have been demonstrated through the playing of the game and a form based feedback. Users have provided feedback on the game.

Centre Mark Awarded: 10

Total Section Mark: 20

Documentation

Documentation **Max Mark: 10**

There is little documentation provided. There is no on screen help, but there is some validation messaging that gives some support when a wrong action occurs. The user guide is very brief and does nt always cover everything expected.

Centre Mark Awarded: 4

Total Section Mark: 4

Evaluation

Discussion of the degree of success in meeting the original objectives **Max Mark: 4**

There is a brief return to the original requirements and mention of if they had been completed or not. However, there is no real "depth" to the evaluation.

Centre Mark Awarded: 2

Evaluate the user's response to the system **Max Mark: 3**

End user feedback is included as part of the testing checks/form that was submitted. The end user provides some feedback on the system, based on game play and time spend using the game. The end user has indicated some faults and development that has been reflected on, in part, by the candidate.

Centre Mark Awarded: 2

Desirable Extensions **Max Mark: 3**

The candidate identidies both positive and negatives within the system, and there is evidence that he has thought about some potential developments. These developments have not necessarily been developed to the point of identifying solutions however.

Centre Mark Awarded: 2

Total Section Mark: 6

Overall Project Mark: 50

Candidate's Work

Text based game project

A-level Computing F452

Section 1 – Definition, Investigation and Analysis

Problem definition

Recently I have noticed a lack of text based games in the market due which I believe is due to them not being as appealing as other games with graphical interfaces and more intuitive gameplay. After talking to friends and looking online it seems as if this is true. Judging by an online chart, action games account for 31.9% of games sold in 2013 with shooters in second with 20% and sports games in third with 12.7%[1]. TBGs do not appear in this particular graph leading me to believe that either the sales are so miniscule that they really don't matter or that there just aren't any TBGs on the market. I can imagine this has left TBG fans disappointed. However, in the past few years retro gaming has become more popular with games such as Shovel Knight[2] and Organ Trail[3] that play with the feeling of nostalgia to appeal to older gamers but can still hold up in today's market. This leads me to believe that TBGs could be an untapped market that simply hasn't been revisited. Text based games didn't really have much wide spread appeal due to computers being so expensive but now 56% of people on Earth own smartphones which are perfectly capable of playing games. As the game industry has grown, the target audience has also grown making most modern games are developed to appeal to the masses rather than a niche market which has led to less profitable/popular genres being phased out. I am planning to have an end user that likes to play text-based games with an age range of 16-35 mainly consisting of males. According to research, the average gamer is 31[4]. Whilst I want the game to appeal to people who played TBGs when they were younger and want to revisit the genre, I also want to introduce younger players to the text based game experience.

My challenge is to create a text based game that will interest and entice current gamers.

Past and current systems

This genre of games usually has a higher focus on storytelling and description through text due to there being no graphical interface. They are generally played by inputting commands into a command line to interact with the game. This is potentially user unfriendly or unintuitive as it requires the user to have prior knowledge of commands and any typing errors would not be accepted. However, this method of input allows for many different options within the game as the only limits to input are the amount of accepted commands within the game.

These games were more common from the 70s to the 90s due to not being too demanding in terms of processing power. They could be played on low spec machines and mostly required only a keyboard to play. Text based games are much rarer these days as hardware has gotten much more powerful allowing for much more demanding games. Newer games are also now much more intuitive in the way that they introduce players to the games mechanics.

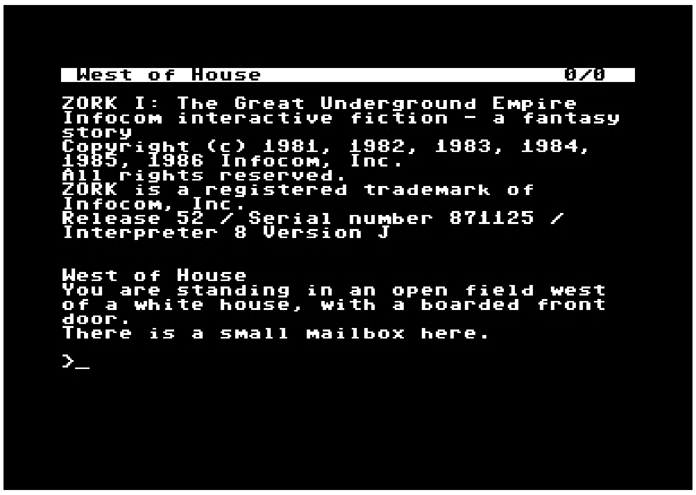
Referring back to the chart from earlier, it seems current gamers tend to be more likely to play action and FPS games than other genres. I think these games are popular as they are relatively easy to understand and can be played both casually and competitively. They are generally fun for both short and long periods of time. They also benefit from being visually appealing. I think In order for my game to fit in with the current market I will probably have to revise the genres mechanics to make them seem less dated and make them more interesting to the current gamer.

There are very few newer text based games currently available. Heroes Rise: The Prodigy is a TBG released in 2012 on Steam. The game uses only text, however rather than having a command line interface where you can input whatever you want, inputs are done by picking from a list of choices. The game has received many positive reviews and is part one of a trilogy.

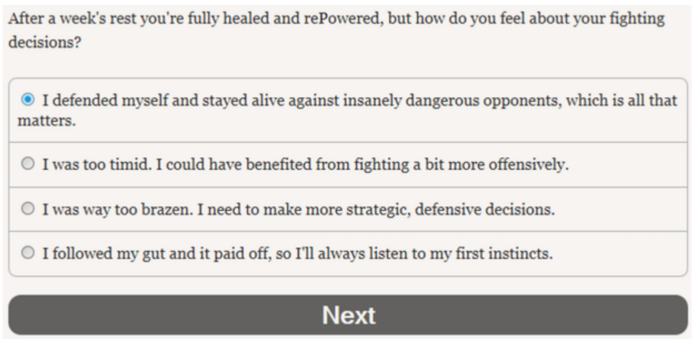
Below is an image of Zork 1 which is one of the earliest text-based games, released in 1980.

[1] http://venturebeat.com/2014/04/29/gaming-advocacy-group-the-average-gamer-is-31-and-most-play-on-a-console/
[2] http://store.steampowered.com/app/250760/?snr=1_7_15__13
[3] http://store.steampowered.com/app/233740/?snr=1_7_15__13
[4] http://venturebeat.com/2014/04/29/gaming-advocacy-group-the-average-gamer-is-31-and-most-play-on-a-console/

Data Forms

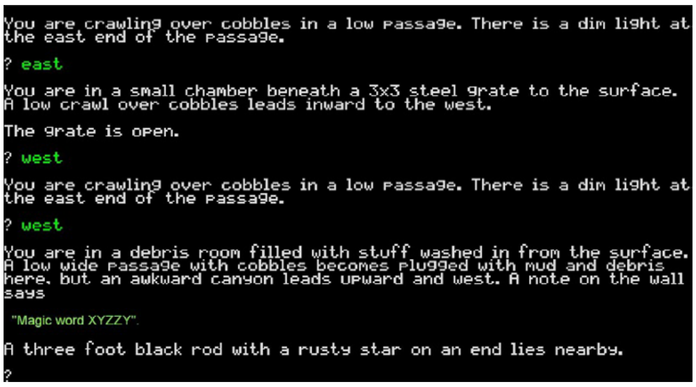


```
 West of House                               0/0
ZORK I: The Great Underground Empire
Infocom interactive fiction - a fantasy
story
Copyright (c) 1981, 1982, 1983, 1984,
1985, 1986 Infocom, Inc.
All rights reserved.
ZORK is a registered trademark of
Infocom, Inc.
Release 52 / Serial number 871125 /
Interpreter 8 Version J

West of House
You are standing in an open field west
of a white house, with a boarded front
door.
There is a small mailbox here.

>_
```



After a week's rest you're fully healed and rePowered, but how do you feel about your fighting decisions?

- [✓ I defended myself and stayed alive against insanely dangerous opponents, which is all that matters.]
- I was too timid. I could have benefited from fighting a bit more offensively.
- I was way too brazen. I need to make more strategic, defensive decisions.
- I followed my gut and it paid off, so I'll always listen to my first instincts.



```
You are crawling over cobbles in a low passage. There is a dim light at
the east end of the passage.
? east
You are in a small chamber beneath a 3x3 steel grate to the surface.
A low crawl over cobbles leads inward to the west.
The grate is open.
? west
You are crawling over cobbles in a low passage. There is a dim light at
the east end of the passage.
? west
You are in a debris room filled with stuff washed in from the surface.
A low wide passage with cobbles becomes plugged with mud and debris
here, but an awkward canyon leads upward and west. A note on the wall
says
 "Magic word XYZZY".
A three foot black rod with a rusty star on an end lies nearby.
?
```

Variable	Data types	From?
Output	String	Game
Input	String	User
Direction	String	Game/User
Magic word	String	Game/User

Investigation and analysis

Types of research

There any several different research types that I could use. Here are some benefits and drawbacks of each:

Questionnaires

With questionnaires, each person answers the same questions and if the respondent is anonymous, more honest answers may be provided. However the questionnaire is only as good as the questions being asked and may be difficult to allow people to expand on their questions. Questionnaires also have to be printed and distributed requiring a lot of time and resources.

Online surveys
Online surveys can be easily distributed to a large group and allows for many people to be involved in the decision making process. Some online surveys will automatically compile the data received into an easily read graph or other form. However it is prone to error and it is possible that questions could be misinterpreted. As they are online they use less resources than questionnaires.

Face to face interviews
Face to face interviews have a high response rate, give you immediate feedback and allow the interviewer to tailor the discussion to the individual. However it requires good interviewing skills and due to the flexibility of the interview results may result in inconsistencies in the results. It is the slowest method of data collection and analysis as it is usually done one person at a time.

Observation
Observation can be combined with other methods of data collection and generates relevant and quantifiable data that can give an in depth look at how the user interacts with the software and can identify persistent problems with the system. The observer can observe multiple people at once. However it requires a skilled observer and if a group is aware that they are being observed then their behaviour may be affected or they may feel uncomfortable.

For my research I will be using online surveys as they allow me to gather data from a large amount of people quickly and easily which will be useful as my system is being developed for an audience rather than a single end user. I will also be able to send the surveys to a wide range of people such as friends and schoolmates. Google forms puts responses from the survey into a either a form or graph making analysis of the results much easier and faster.

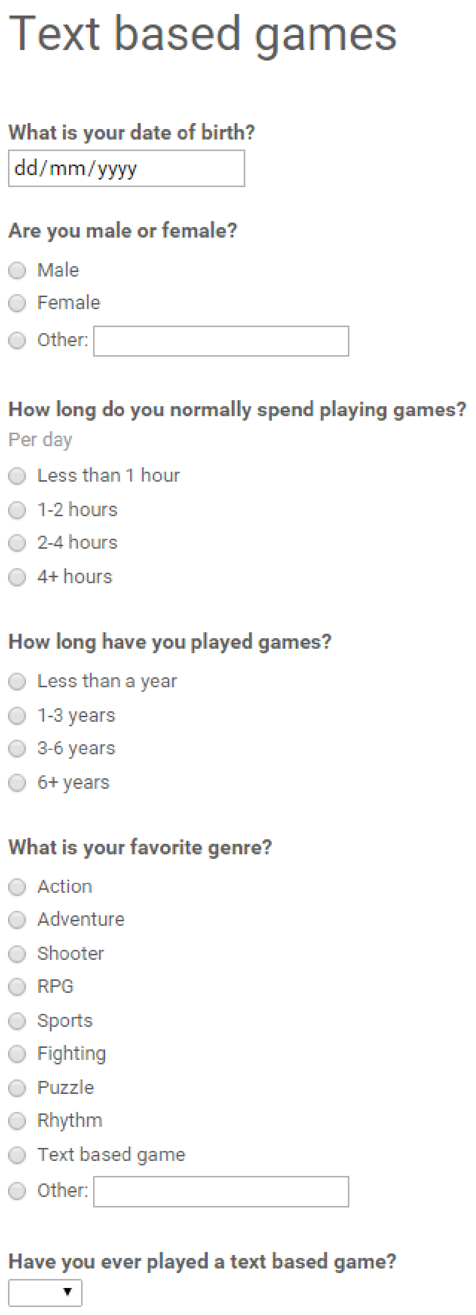
Research
As this game is not being created for a single end user I will send out questionnaires to select students in my college that are within my target age range and target demographic. Some examples of questions I could ask are;

- How old are you?
- Are you male or female?
- How long do you normally spend playing games?
- How long have you played games?
- What is your favourite genre?
- Have you ever played a text based game?
  - If so, how often do you play text based games?
  - What do you like about text based games?
  - What do you dislike about text based games?
- If not, why?
- What features would you like to see in a text based game?
- Do you prefer to play games online (Multiplayer) or offline (Single player)?

I am also going to observe others playing text based games to get an idea of how others play games.

Questionnaire results
I sent out 19 questionnaires and all recipients replied however some recipients have not filled out the entire survey.

Below is an image of the questionnaire.



Text based games

What is your date of birth?

dd/mm/yyyy

Are you male or female?

○ Male
○ Female
○ Other:

How long do you normally spend playing games?

Per day

○ Less than 1 hour
○ 1-2 hours
○ 2-4 hours
○ 4+ hours

How long have you played games?

○ Less than a year
○ 1-3 years
○ 3-6 years
○ 6+ years

What is your favorite genre?

○ Action
○ Adventure
○ Shooter
○ RPG
○ Sports
○ Fighting
○ Puzzle
○ Rhythm
○ Text based game
○ Other:

Have you ever played a text based game?

Here are the results:

How what is your DOB?

1. 09/04/1996
2. 24/09/1997
3. 02/12/1997
4. 30/05/1996
5. 07/01/1998
6. 19/07/1997
7. 12/08/1997
8. 23/03/1996

9. 13/03/1997
10. 29/06/1997
11. 20/03/1995
12. 17/12/1996
13. 01/09/1997
14. 05/01/1988
15. 27/05/1980
16. 20/11/1993
17. 30/07/1998
18. 09/04/1990
19. 24/06/1996

Are you male or female?

1. Male (10)
2. Female (9)

How long do you normally spend playing games?

1. 4+ hours (6)
2. 2-4 hours (11)
3. 1-2 hours (2)

How long have you played games?

1. 6+ years (16)
2. 3-6 years (3)

What is your favourite genre?

1. Action (4)
2. Adventure (3)
3. Shooter (5)
4. RPG (2)
5. Sports (1)
6. Fighting (2)
7. Puzzle (1)
8. Rhythm (1)
9. Text based game (0)
10. Other: (0)

Have you ever played a text based game?

1. Yes (16)
2. No (3)

If not, why?

1. Looks boring.
2. Uninterested.
3. Never had the chance.

If so how often do you play text based games?

1. Once a year (13)
2. Once a month (3)
3. Blank (3)

What do you like about text based games?

4. They're a way to pass the time when I'm extremely bored. Honestly, I would rather read a book.
5. Wide amount of possibilities to choose from.
6. It causes the player to think and make decisions appropriate to the situation.
7. Range of choice.
8. The role play aspect you can get from them.
9. Lots of focus on the story line and text based games are usually more dynamic.
10. I enjoy using my imagination to picture the described settings and scenarios.
11. The ability to choose and have multiple outcomes depending on the choices you make.
12. nothing much, just helps me to keep track of my touch typing skills
13. Blank
14. I like the that the world is described through text as it leaves it up to my imagination as to what everything looks like.
15. Easy to understand how to play.
16. Good pastime
17. There is generally a lot of attention to detail when describing surroundings.
18. Nothing
19. Blank
20. Accessibility. You don't need to remember many button combinations, you just need to know how to type.
21. Blank
22. I like that there is an element of choice

What do you dislike about text based games?

1. I find them dull for the most part, unless they tackle situations I am interested in.
2. Reminding me my imagination is bad.
3. They're only text.
4. No images.
5. The repetitive nature of the story.
6. They are text based games.
7. The rigid structure of text-based games often restricts me from getting the most out of my in-game decisions, with the exception of a few. Some require a lot of work as the actions have to be typed in their own rules. You're not allowed to say "Use Key." You must specify "Use Golden Abyss Key on West Golden Key Door and Unlock Golden Key Door and Open Golden Key Door."
8. blank
9. Very dependent on the story.
10. It's boring and plain, there's nothing to it.
11. Needs an interesting premise, otherwise the game just falls apart.
12. Tiring looking at text all the time.
13. Not very rewarding.
14. Not much in the way of gameplay.
15. No visuals.
16. Too much text.
17. Pretty boring and railroady.
18. Not very interesting.
19. Choices usually don't make much difference.

What features would you like to see in a text based game?

1. A Dragon Age style list of options, rather than having to guess at what to type.
2. Character portraits, fun.
3. More than text.
4. Some images (maybe in ASCII) to give a better idea of the situation. Spell checker built in.
5. A good story. Colours text and/or background (Not black and white). Pictures/images of characters in the game (to help with role-play).
6. Different endings, different options, a lot of variety and diversity, a lot of choice and maybe some graphics.
7. More variation in options, easier-to-type actions as opposed to long strings on sentences.
8. Maybe a way to make it more fun.
9. Blank.
10. GUI.
11. Custom scenario creator.
12. Less text and a graphical interface.
13. A combat system.
14. Variety in gameplay.
15. Visuals.
16. Some images to go with the text.
17. Other things to do besides the main story that the games is telling.

18. Interesting mechanics
19. Choices that significantly affect the progression of the story

Would you prefer a game that uses only text or a game with both a GUI and text?

1. Text (0)
2. GUI and Text (19)

What do you like about modern games that aren't available in TBGs?

1. Freedom of movement as opposed to restrictive parameters and invisible walls. Modern games also tend to feature finer details in their models. 'Planescape: Torment' was an excellent TBG for its time on account of its excellent story, freedom and artwork.
2. There's more than text.
3. There's a story and a quest/objective to complete
4. You can understand a character by seeing their reactions to events whereas in a TBG this is not the case as they may not describe the player's character and only give them options of direction and significant landmarks.
5. More possibilities in regards to control, gameplay etc.
6. User interfaces
7. Skill based combat systems rather than turn based
8. The variety and amount of content
9. Graphics
10. Intuitivity of control
11. There's a story and a quest/objective to complete
12. Online multiplayer
13. Nothing (7)

Analysis of feedback
From my feedback I can tell that most people have at least played a text based game once in their life, however the results also show that those who have played them do not play them frequently, mostly once a year.

Gameplay
People generally like the focus on story and the thought that must be put in when choosing a path or figuring out puzzles. This could mean that there should be a focus on the story, puzzle and choice elements of the game. Wanted features include a graphical interface, interesting gameplay mechanics, character portraits, a combat system and an interesting story. These features are possible to implement with the time I have however some wanted features are not possible for me to implement such as multiplayer and meaningful choices that heavily impact story progression

UI and visuals
Lack of gameplay and lack of any graphical interface to go with the text are the most commonly brought up problems. Almost all recipients prefer GUI and text over just text. This means that I should probably include visuals in the game. By far the most wanted feature judging by the answers from each question is a graphical interface.

Controls
Highlighted features in modern games that are not in TBGs include intuitive controls, graphics, freedom of movement and variety of content. This could mean that as well as a command line there should also be shortcut keys for common actions to make it more intuitive.

Initial ideas from results
From my results I have compiled a list of ideas based on the results from the questionnaire.

The game will need:

Gameplay
- Variety of things to do.
- Puzzles that require some thought.
- A combat system.
- An emphasis on story.

UI and visuals
- A graphical interface as well as text.
- On screen buttons as well as a command line.
- A portrait showing the characters face (possibly shows emotions or status effects).
- GUI to show Health, level, buttons etc.

Controls
- Shortcut buttons for commonly used commands
- Will use keyboard and mouse

Second questionnaire
Now that I have compiled the results of the first questionnaire and produced a list of user requirements I need to find out if the end user agrees with them and ask some more in depth questions related to the answers from the previous questionnaire and technical questions.

I sent this questionnaire back to the people who answered the first questionnaire and got full replies.

Are you happy with these user requirements?

1. Yes (15)
2. No (4)

If not, is there anything you would add to the list?

3. When someone says there are meaningful choices, they usually aren't. How will they be meaningful, and are they more consequence actions then actual choices? How can a Text based game be fast paced even though typing speeds differ by person to person?

Do you think there should be multiple difficulty settings or one set difficulty?

1. Multiple (8)
2. Single (11)

Do you think the game should focus mainly on combat, puzzles or story?

1. Combat (8)
2. Puzzle (5)
3. Story (6)

What information do you think should be displayed on the game GUI?

1. Health, inventory, notepad (to write down notes and clues), General actions to save time typing them each time (E.G. movement, look, take, put, interact). No mini-map, the only map should be one the player has to stop and bring it up.
2. Health, mana and minimap.
3. Score, time, currency, health.
4. Basic info such as health and mana
5. Health, buttons and command line.
6. Health
7. There should be a save button, text box, character portrait and command line.
8. Health, mana, text box
9. Inventory, health, pause menu button
10. Health and mana bar, inventory and text I/O
11. There should probably be a health and mana bar and an inventory box
12. Health, minimap and inventory.
13. Compass, health, mana, stamina, stats, inventory, text box
14. Health bar, mini map, menu button
15. Player stats, enemy stats, items
16. Map, abilities, items, enemy health
17. Health, buttons for movement and combat, inventory, map/compass
18. Objectives, health, equipment, map with layout of rooms
19. Health, mana, stamina, items, money, current task

Roughly how long would you want each level to take?

1. 30 minutes, could differ depending on the puzzles.
2. 15-20 minutes.
3. 2~5 minutes.
4. Around 5 mins
5. 5-10 mins
6. A few minutes
7. No more than 10 minutes.
8. Between 10 and 15 minutes
9. Couple mins
10. Less than half an hour.
11. 10 minutes
12. <5 Mins
13. Somewhere around 10 minutes
14. Between 5 and 8 mins.
15. 3 minutes.
16. 5 minutes
17. Around 5 mins.
18. 5~15 minutes
19. 10 to 20 minutes

What sort of obstacles would you like to see in the game?

1. Pattern puzzles, pressure sensitive pads, Enemies that take thought to overcome.
2. Whatever you want.
3. I would say a locked door or a chasm the player requires a special ability to pass would work.
4. Generic monsters; skeletons, goblins etc.
5. Pitfalls, enemies, traps
6. Hostile creatures, traps, illness
7. Decisions resulting in a combat scenario.
8. Monster ambushes
9. Spike pits, Key doors
10. Locked doors/gates
11. Skeletons, zombies
12. Doors that can only be open by solving a puzzle.
13. Puzzles
14. Locked doors, dead ends
15. Traps, enemies
16. Riddle/Puzzle activated doors, weak flooring
17. Blowpipes, boulders, timed doors
18. Locked doors/gates, fake walls, enemies
19. Rats, goblins etc.

Would you prefer turn based or real-time combat?

1. Turn based (12)
2. Real time (7)

Would you prefer 2D or 3D graphics?

1. 2D (14)
2. 3D (5)

Would you prefer a manual save or auto save feature?

1. Manual (11)
2. Auto (8)

What do you think should be included in the options menu?

1. Volume, resolution, sensitivity
2. It should include sound control and resolution
3. Sound, video, control options
4. Video and Sound settings
5. Graphics options, key bindings, mouse sensitivity
6. Video, control and sound settings
7. Graphics settings

8. Key bindings, mouse sensitivity, resolution, render quality, music and SFX volume sliders
9. Sound and video
10. Game and control options
11. Graphics, music, control settings
12. Resolution, texture/model quality, sounds, controls
13. Screen, music and game settings
14. Video, sound, control and game
15. Graphics, sound, key bindings
16. Sound, controls
17. Controls, sound and graphics
18. Graphics, sound and controls
19. Video(graphics, resolution), sound(Music, SFX) and control options(Key rebinding, mouse sensitivity)

Would you like the option to choose your gender?

1. Yes (9)
2. No (10)

Do you think that there should be merchants that sell various items?

1. Yes (16)
2. No (3)

Would you prefer being able to name the player character or that the player character have a set name?

1. Custom (15)
2. Set (4)

Analysis of second survey
As most were happy with the current requirements I decided to not remove any and only add to the list however the majority of recipients preferred a turn-based approach to combat rather than a real time system so I have changed the list to reflect this.

Gameplay
There was a lot of variation in the responses regarding the length of each level but 5 and 10 minutes showed up the most so I decided that would be the target length for each level. It seems that the majority would prefer a single set difficulty so I decided that the game will be an intermediate difficulty. Players seem to prefer naming their character themselves so I will include that in the game. The recipients seem to want the game to focus mostly on the combat with story coming second and puzzle solving coming last. This clashes with an observation I made in the analysis of the first survey, however here they have directly stated that they would like a focus on combat so I will go with that.

UI and visuals
Each recipient had similar ideas for information to be shown to the player and that were relevant so I decided to add any that made sense such as health and inventory. Overall the recipients seemed to favour a manual save over an automatic one, so a save button will be included on the UI.

Controls
It was requested that buttons should be available on the GUI that are shortcuts to common commands to avoid having to type in the same commands over and over again.

Second list of ideas
I have revised the initial list of ideas to include results from the second questionnaire.

The game will need:

Gameplay
- Variety of things you can do.
- To be able to set the name of the character.
- Puzzles that require some thought.
- To have a turn based combat system.
- Merchants that sell various helpful items.
- Various obstructions such as locked doors and pitfalls.
- Levels that take around 5-10 minutes to complete.
- Have an overall intermediate difficulty.

UI and visuals
- 2D graphics.
- A graphical interface as well as text.
- A portrait showing the characters face (possibly shows emotions or status effects).
- A GUI to show information, buttons etc.
- Information such as health, mana, currency, minimap, inventory and general actions to be displayed on the GUI.

Controls
- On screen buttons as well as a command line.
- Will use keyboard and mouse.

Other
- An options menu containing audio, video and control options.

Final list of requirements
I have compiled my observations into a final list of requirements. It is basically my list from the second questionnaire with some alterations based on my ability and time.

Gameplay
- Variety of things you can do.
- To be able to set the name of the character.
- Puzzles that require some thought.
- To have a turn based combat system.
- Merchants that sell various helpful items.
- Various obstructions such as locked doors and pitfalls.
- Levels that take around 5-10 minutes to complete.
- Have an overall intermediate difficulty.

UI and visuals
- 2D graphics.
- A graphical interface as well as text.
- A GUI to show information, buttons etc.

- Information such as health, currency, inventory and general actions to be displayed on the GUI.

Controls
- On screen buttons as well as a command line.
- Will use keyboard and mouse.

Other
- An options menu containing video and control options.
- A death screen for when the player is defeated in combat.
- A menu screen with buttons for new game, load game, options and exit

Software and Hardware requirements
For the end user/running this system, these will be the requirements:

Software	Reason
Any recent version of Windows, MacOS or Linux	A platform for the program to run.

Hardware	Reason
Around 1GB storage space	To be sure that there is enough space to store the program locally.
1.6 GHz or faster processor. 1 GB of RAM (1.5 GB if running on a virtual machine). 10 GB (NTFS) of available hard disk space. 5400 RPM hard drive. DirectX 9-capable video card running at 1024 x 768 or higher display resolution.	I was unable to find minimum requirements for running a Visual Basic program so I will assume it is the same as the minimum for Visual Studio itself.

For creating the game, these will be the requirements

Software	Reason
Any recent version of Windows, MacOS or Linux	A platform for me to work on the program
Visual studio	Software that I'm going to use to create the program

Hardware	Reason
1.6 GHz or faster processor. 1 GB of RAM (1.5 GB if running on a virtual machine). 10 GB (NTFS) of available hard disk space. 5400 RPM hard drive. DirectX 9-capable video card running at 1024 x 768 or higher display resolution.	Minimum requirements for running visual studio.

I know that the school computers have at least the minimum recommended specs, so they are a suitable platform for developing the program.

Section 2 – Design and Testing

Objectives

Though I already have a list of requirements for the game, I have decided to adapt this list to give myself a clear indication of what is required and desired of the game interface. After I have finished these designs I will present them to my end users and get feedback to make improvements.

Inputs

Main menu

- A button for starting the game without loading a previous save file.
- A button for loading the game using a previous text file.
- A button for accessing the options menu.
- A button for exiting the application.

Game

- Text input to allow the use of specific commands.
- Shortcut buttons for commonly used commands such as moving and attacking.
- A button for accessing the pause menu.

Pause menu

- A button for resuming the game.
- A button for saving current progress.
- A button for accessing the options menu.
- A button for exiting to the main menu.

Options menu

- Buttons for changing the size of the window between 2-3 pre-set sizes.
- Text input for changing the save location on the hard drive.
- Text input for changing the players name.

Outputs

Main menu

- Image box for displaying the logo of the game.

Game

- Image box to display the game screen.
- The image box will show different images to suit what is currently happening in the game.
- Text output to show story and events in the game.
- A text box showing layer stats such as health, money and status effects.
- A text box for displaying the inventory of the player.

Aesthetics

Main menu
- Red background.
- White Buttons with black text.
- Logo at the top of the screen.

Game
- Red background.
- White Buttons with black text.
- Reasonably different sizes between window size options.
- Game screen and text output above inputs, inventory and stats.

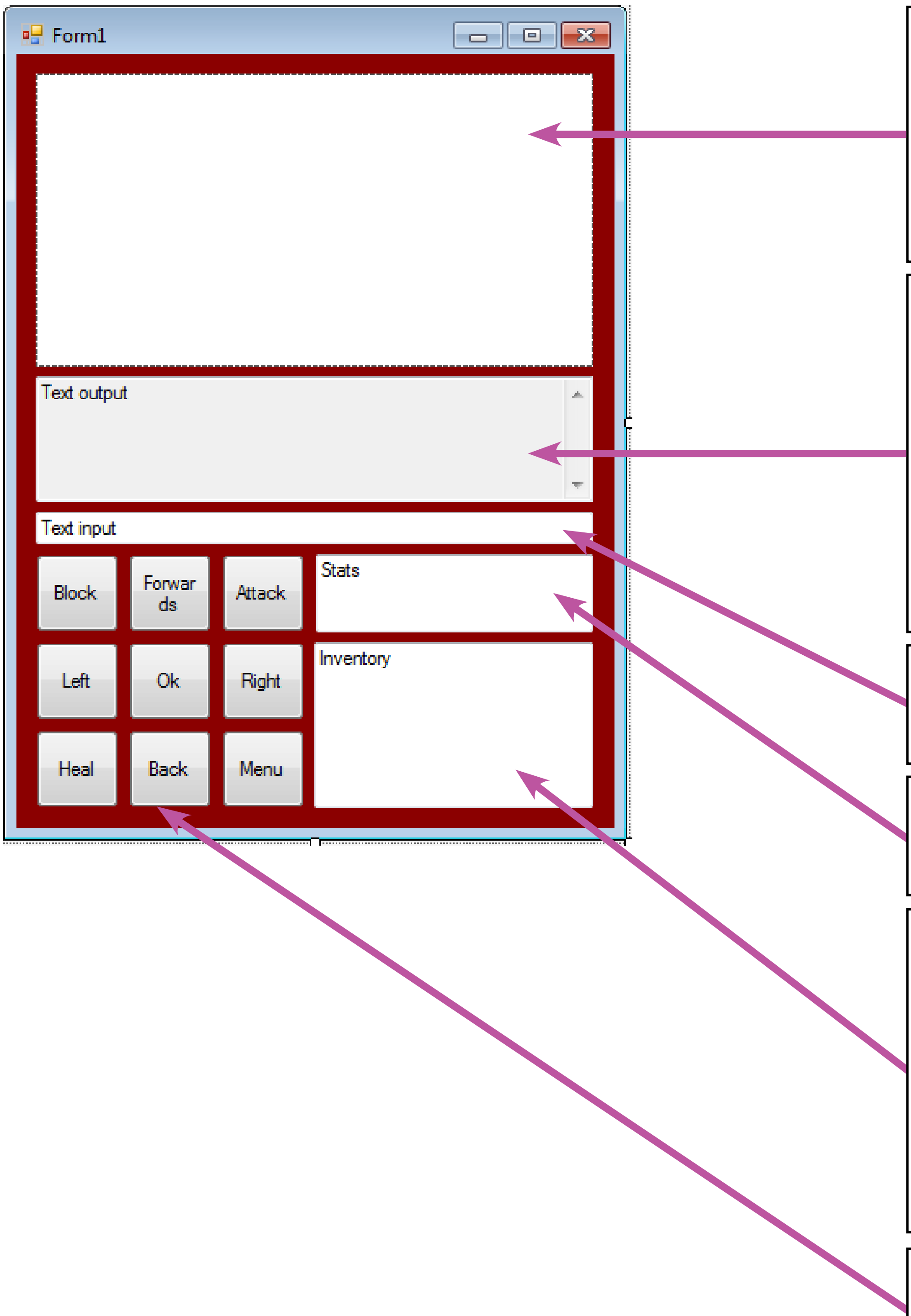
Pause menu
- Red background.
- White Buttons with black text.
- Options menu
- Red background.
- White Buttons with black text.

Processing

Game
- The layout of the rooms will be stored in an array. When the player moves in a direction, the game will get the location of the variable adjacent in the array in the direction the player specifies. The Image box loads an image based on the location in the array.
- Current items held by the player will also be stored in an array and constantly output to the inventory box.
- Descriptions of rooms will be stored as strings in an array and output to the text output box.
- Health and money will be stored as integers in an array and constantly output to the stat box.

Interface designs

Game interface



Display – Shows the game screen using an image file that is accessed depending on which room the player is in. The game screen is the largest module on the interface to keep focus on the game.

Text output – Displays the text that shows descriptions, combat statistics and story. I have placed this directly underneath the game display so that the player can quickly read the text and also see what's happening in the game. It also includes a scroll bar so that the player can look back on previous events.

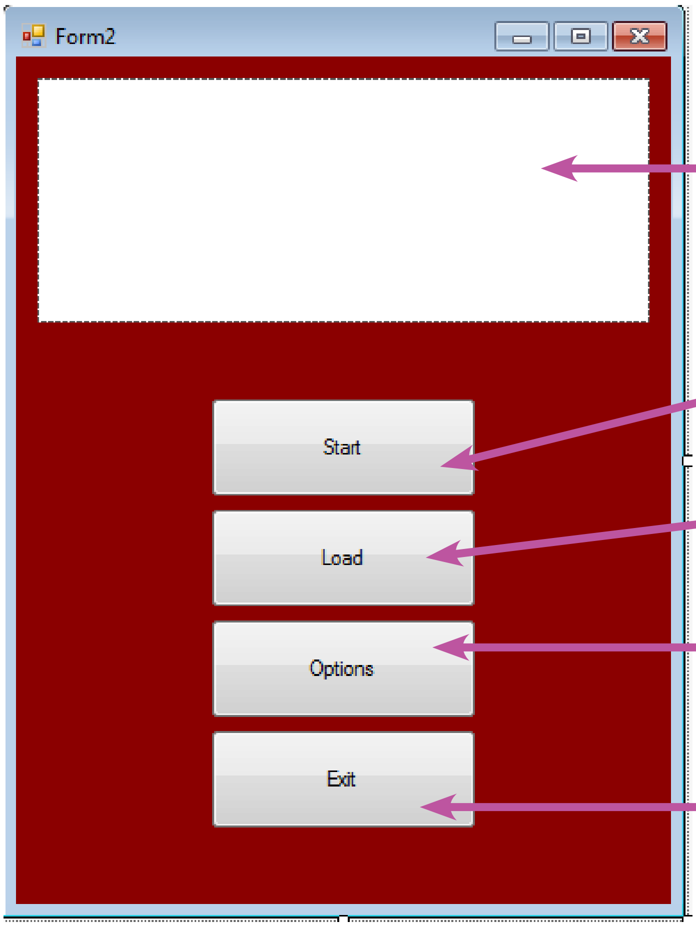
Text input – Secondary input used for accessing specific actions/commands.

Stats – Shows current health and currency. Maybe also status effects such as poison, sleep etc.

Inventory – Shows items in the game currently held by the player such as health potions, keys etc. The inventory and stats boxes are close to the shortcuts and text input so that the player can look at them and quickly make an action depending on their current situation.

Shortcuts – Buttons for quick and easy use of common commands to avoid repetitive typing.

Main menu interface



Picture box – Shows logo for the game.

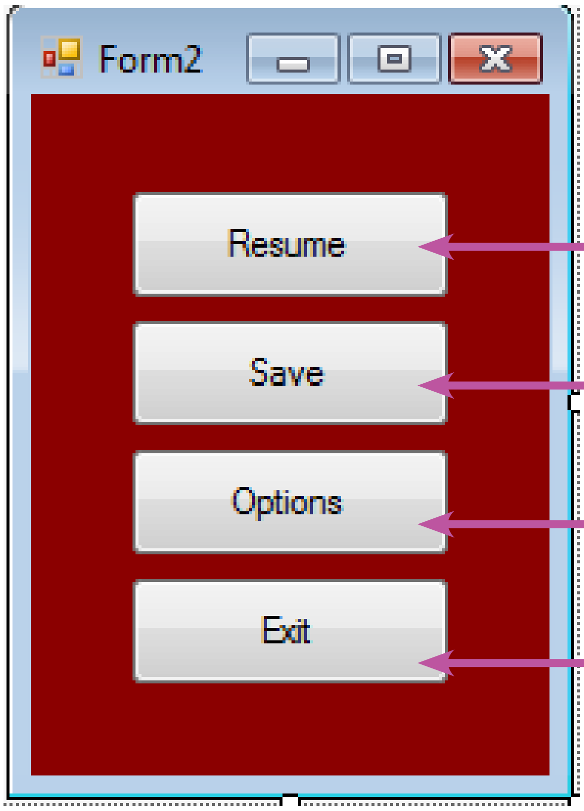
Start button – Begin a new game without loading a previous save.

Load button – Load a previous saved game and start from where you left off.

Options button – Access the options menu where the player may alter certain settings.

Exit button – Close the application.

Game menu interface



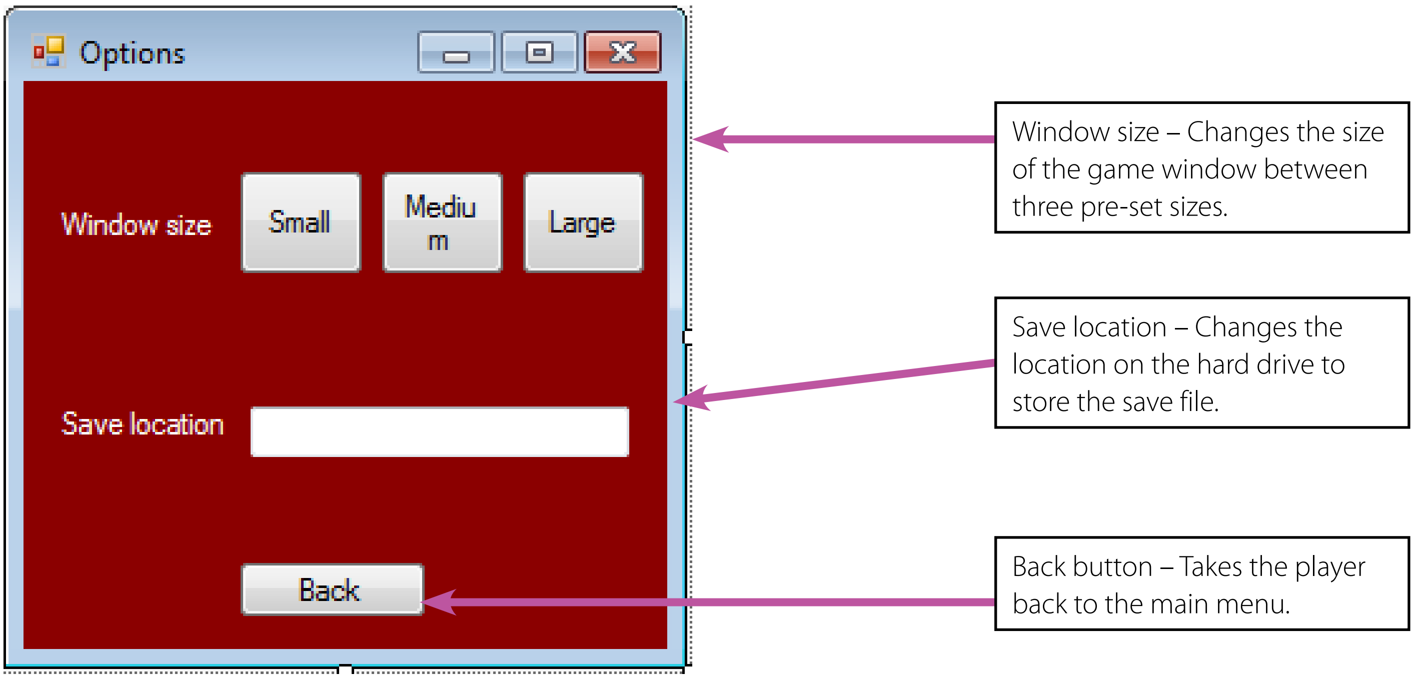
Resume button – Continue playing the game.

Save button – Save current progress to a file which can then be loaded later.

Options button – Accesses the options menu.

Exit button – Exits to the main menu.

Options menu interface

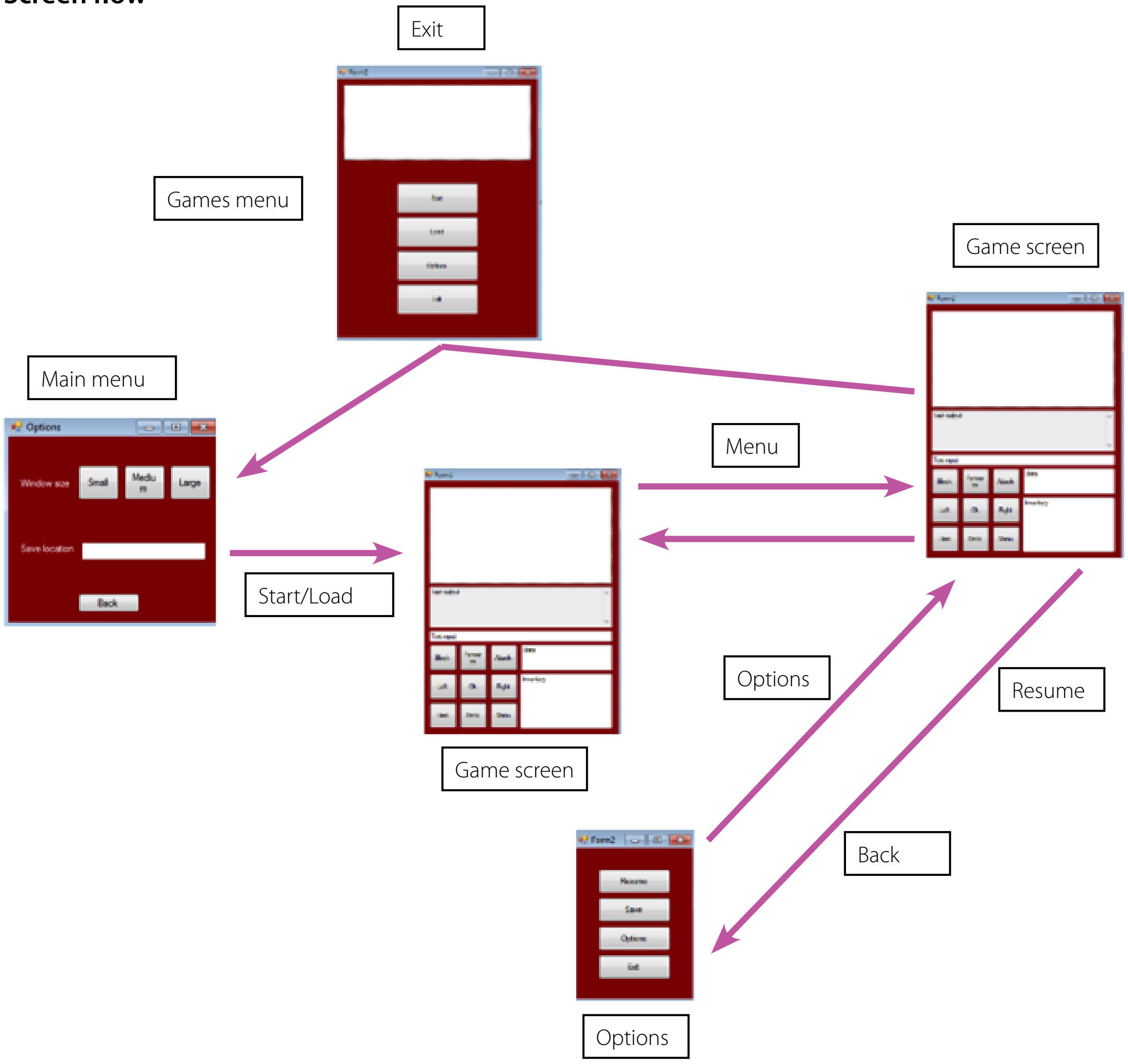


Window size – Changes the size of the game window between three pre-set sizes.

Save location – Changes the location on the hard drive to store the save file.

Back button – Takes the player back to the main menu.

Screen flow



End user feedback

Before continuing, I sent out a survey to get feedback from my end user on the current screen designs.

Results

Do you think that the current game screen design is suitable for the program?
Yes (19)
No (0)

Is there anything you would add/change about the game screen design?
No (19)

Do you think that the current main menu screen design is suitable for the program?
Yes (19)
No (0)

Is there anything you would add/change about the main menu screen design?
No (19)

Do you think that the current game menu screen design is suitable for the program?
Yes (19)
No (0)

Is there anything you would add/change about the game menu screen design?
No (19)

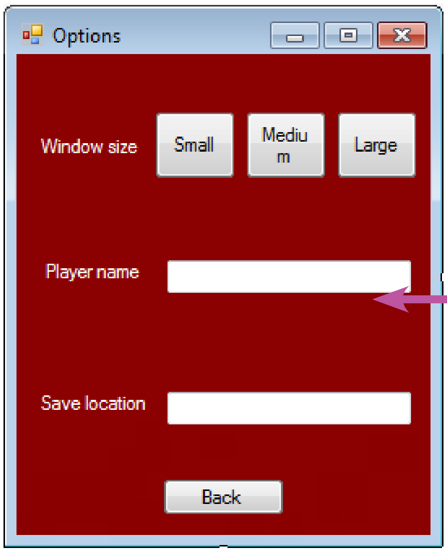
Do you think that the current options screen design is suitable for the program?
Yes (18)
No (1)

1. There needs to be an area where you can input your name.

Is there anything you would add/change about the options screen design?
No (19)

Changes

Based on the feedback above I have made changes to the design of the options menu. It now has an area where the player can input their name.



Player name – Sets the name that the player will be referenced by.

Variables and data structures

Variable	Type	Identifier	Description
Health	Integer	health	The amount of health the player currently has.
Currency	Integer	money	The amount of gold the player currently has.
Password	String	password	The word used to solve a riddle to open a door.
Keys	Integer	keyCount	The amount of keys the player currently has.
Health potions	integer		
Player name	String	playerName	The name of the player. The game will sometimes reference the player using this.
Room layout	Array/Integer	rooms	An array containing the location of each room.
Sword	String	playerSword	Holds the name of the sword currently equipped. When a new sword is acquired the old sword will be overwritten.
Shield	String	playerShield	Holds the name of the shield currently equipped. When a new shield is acquired the old shield will be overwritten.
Armour	String	playerArmour	Holds the name of the armour currently equipped. When a new armour is acquired the old armour will be overwritten.
Sword damage	Integer	swordDamage	The amount of damage that the sword does based on the current sword equipped.
Shield negation chance	Integer	shieldBlock	The chance that the shield has to block based on the current shield equipped.
Armour damage reduction	Integer	armourDamRed	The amount that damage received is reduced by based on the current armour equipped.
Inventory	Array/Integer	inventory	Holds the keys and health potions that the player holds.

Outline of story and basic game structure

The basic outline of the story is that the player has entered into an event to win huge amounts of money. However to receive this money, the player must survive through a dungeon-like maze filled with deadly traps, creatures and puzzles. Along the way you may come across other entrants who will either trade with you or fight you.

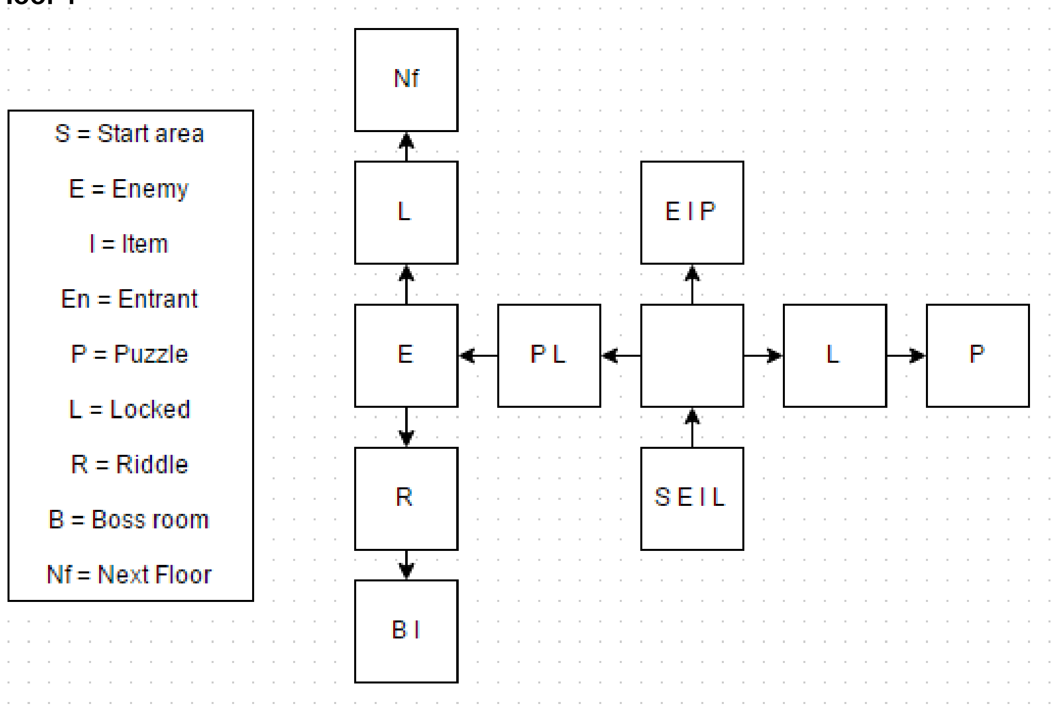
You wake up in a room with a bed and table. On top of the table is a tray with some food for restoring health and a key which will open the door to the maze introducing the player to the inventory and healing mechanics. There will also be a small harmless creature such as a rat for introducing the combat mechanic. The first room will have three branches. The first branch will lead towards a password door and a lever. The second branch will lead to a combat situation, some food, a rusty sword that slightly increases damage output and a second lever. The third branch leads to a gate which is opened once both levers are thrown. On the other side is a puzzle to figure out the password for the door from the first branch. After passing the password door there will be a combat situation and 2 branches, the first leading towards a riddle door and the second leading towards a locked door. Behind the riddle door will be a mini boss which will drop a key and a suit of rusty armour which will slightly decrease damage received. Using the key on the second branch door will lead to a staircase that takes the player to the second floor.

In the first room there are two branches and an entrant who wants to trade a shiny object and health potions for gold. The player should not currently be able to buy the shiny object. Following the first branch there will be a combat situation and a door with a slot in the middle. Following the second branch there will be an empty room with a fragile wall. Attacking this wall will reveal a bag of money. The player can now buy the object from the other entrant and use it to open the door with the slot. There will be 2 more branches, the first of which will lead to another entrant who will try to fight you, however you can try talking to him to calm him down. If you manage to calm him down he will give you a key piece and offer a trade for a rusty shield which has a random chance to block an attack. Defeating him in battle will give you only the key piece. Taking the second will lead to another riddle door. In the next room there will be three branches, two of which lead to combat situations which both drop a key piece, the third branch needing a key to enter. Once all three key pieces are in possession the player will be able to enter the boss room. Defeating the boss will end the game.

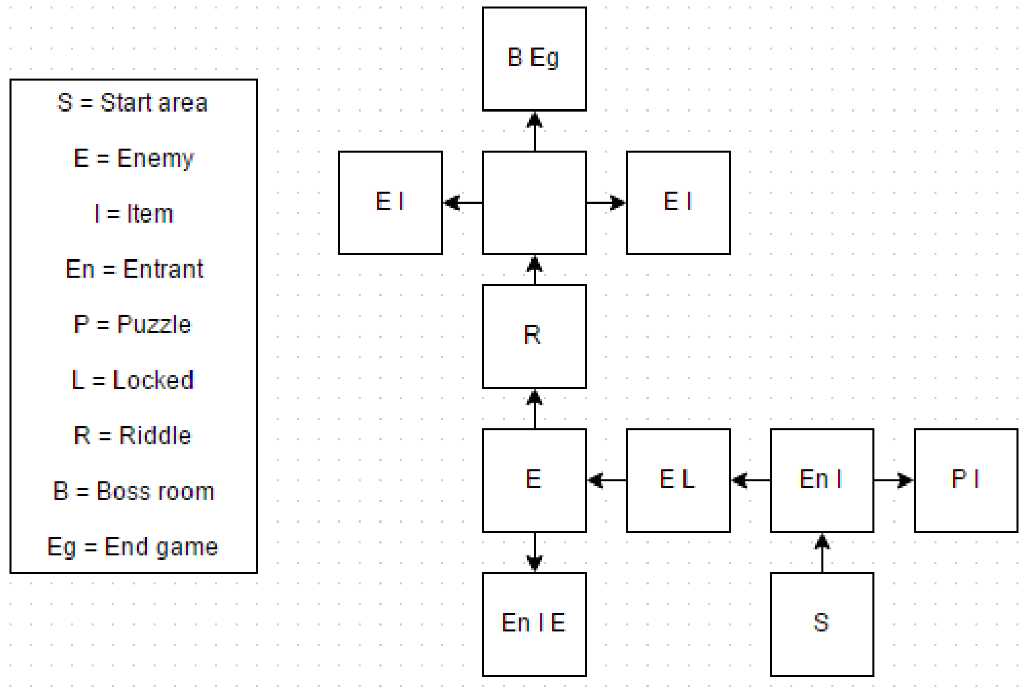
Floor layouts

Using the gameplay outline above I have created rough layouts of the floors which will help when constructing them.

Floor 1



Floor 2



Combat

As combat will be turn based, the structure of the combat will go like this;

1. The player gets to take their turn first.
2. They have the option to attack, heal or defend.
    - Attacking will deal a random amount of damage between 10 and 20 + current sword attack to the enemies' health.
    - Blocking will reduce damage received by half and has a chance to negate damage completely.
    - Healing will consume a health potion and restore around 50 health points.
3. It is then the enemies turn.
4. The AI will mostly attack but will occasionally heal
5. This will repeat until either the player or enemy is defeated.
6. If the enemy is defeated the player will receive 50 gold and in some cases an item.
7. If the player is defeated the game will end.

Movement

Movement will be rather basic. Clicking on a directional button or typing in something like "Move right" will move the player to an adjacent room in the direction specified if there is one available.

Pseudo code

Before I can start coding the program I will need a clear set of pseudo codes so that I can work out how I am going to structure my code and so that I know exactly what module to work on next.

Below I have provided the pseudo code ordered as modules. The modules are combat, movement, puzzles and general. They are currently ordered like this so that I can clearly see which bits of code do what.

Issuing commands

```
Int playerX = 3
Int playerY = 3
String command
Bool loop = true
Bool usePresent = false
string array coordinates [6,6] {A1, B1, C1, D1….} {A2, B2, C2, D2…..}…….

WHILE loop is true
  IF (combatSituation = true)
     RUN combat
  ELSE
  Write "Please enter a command."
  Command = readline

  IF (Command = up AND combatSituation = false)
     PlayerY++
     Write command [playerX, playerY]
     Command = null
  ELSE IF (Command = down AND combatSituation = false)
     playerY–
     Write command [playerX, playerY]
     Command = null

  ELSE IF (Command = left AND combatSituation = false)
     playerX–
     Write command [playerX, playerY]
     Command = null

  ELSE IF (Command = right AND combatSituation = false)
     playerY++
     Write command [playerX, playerY]
     Command = null

  ELSE IF (command = use AND usePresent = true)
     Activate room device

  ELSE IF (command = menu AND combatSituation = false)
     Create single instance of pause menu
  ELSE
     Write "Please enter a valid command."
  END IF
```

Combat

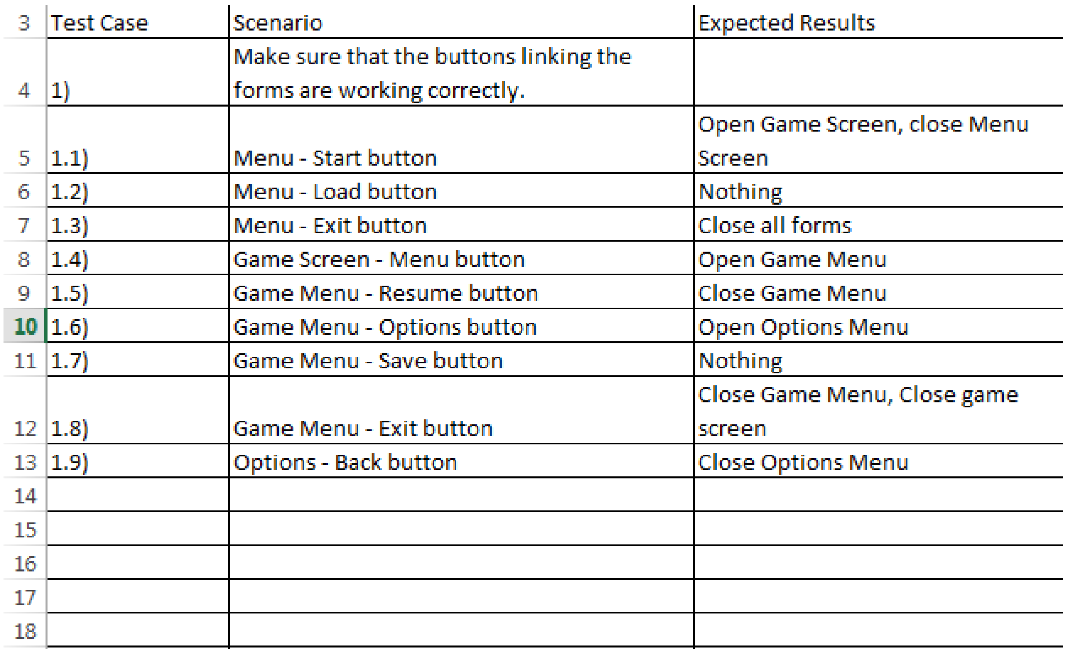
```
WHILE combat Situation is true and enemy Alive is true
  enemy action = random between 1 and 10
  IF enemy action > 1 THEN
     Enemy attack = random between 4 and 10
  ENDIF
  Read line for action
  IF action = attack THEN
     Attack damage = random between 10 and 20
     Attack damage = attack damage + sword damage
     IF enemy health > attack damage THEN
Enemy health = enemy health - attack damage
     ELSE IF enemy health < attack damage THEN
          Enemy alive = false
          Write "Enemy defeated" to text output
     ENDIF
     Write "Enemy damaged for (attack damage)" to text output

  ELSE IF action = block THEN
     Block chance = random between 1 and 20
     Block chance = block chance - shield block chance
     IF block chance = 1 THEN
          Enemy damage = 0
     ELSE
          Enemy damage = enemy damage / 2
     ENDIF
  ELSE IF action = heal THEN
     IF (player max health - player health) < player heal amount THEN
```

```
                Player health = player health + player heal amount
           ELSE IF (player max health - player health) > player heal amount THEN
                Player health = player max health
           Write "(player name) healed for (player heal amount)" to text output
        ELSE
           Print "Please enter an action"
        ENDIF

        IF enemy action = 1 THEN
           IF (enemy max health - enemy current health) < enemy heal amount THEN
                enemy health = enemy health + enemy heal amount
           ELSE IF (enemy max health - enemy current health) > enemy heal amount THEN
                enemy health = enemy max health
           Write "Enemy healed for (enemy heal amount)" to text output
           ENDIF
        ELSE
           IF player health > enemy attack
                Player health = player health - enemy attack
           ELSE IF player health < enemy attack
                Player health = 0
        Write "(player name) damaged for (enemy attack)" to text output
      ENDIF
```

These modules together should allow for the player to navigate through the game, interact with the NPCs and objects and engage in combat. However, these modules are very rough and will need to be built upon when I start coding. These will give me a firm base to start coding the game.

Test Plan

I plan to test the program myself using a test table. User testing will be done by sending out questionnaires along with a copy of the game. Below is an example of the sort of test table that I will use.



Test Case	Scenario	Expected Results
1)	Make sure that the buttons linking the forms are working correctly.	
1.1)	Menu - Start button	Open Game Screen, close Menu Screen
1.2)	Menu - Load button	Nothing
1.3)	Menu - Exit button	Close all forms
1.4)	Game Screen - Menu button	Open Game Menu
1.5)	Game Menu - Resume button	Close Game Menu
1.6)	Game Menu - Options button	Open Options Menu
1.7)	Game Menu - Save button	Nothing
1.8)	Game Menu - Exit button	Close Game Menu, Close game screen
1.9)	Options - Back button	Close Options Menu

Using a test table allows for time efficient testing with minimal hassle and easy readability.

Section 3 – Implementation and Testing

Project

UI and menu navigation

Based on my designs in order to develop this solution I will first need to draw up each room of the game that will be displayed. I will use a base room template at the same resolution as the game screen. Looking at the properties of the game screen, the size of the game screen is 342 by 180 pixels.

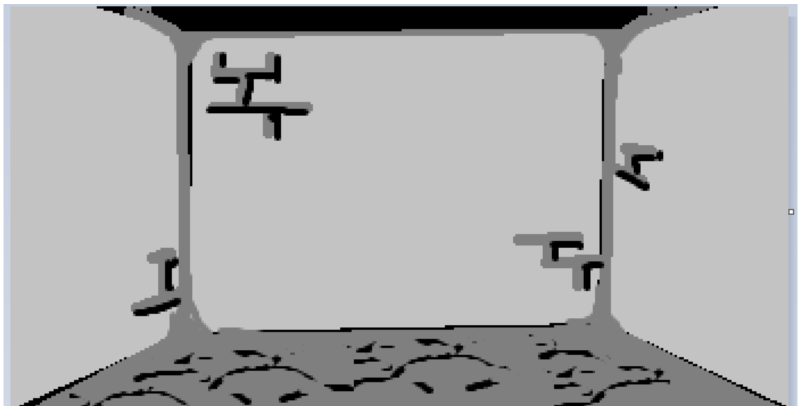
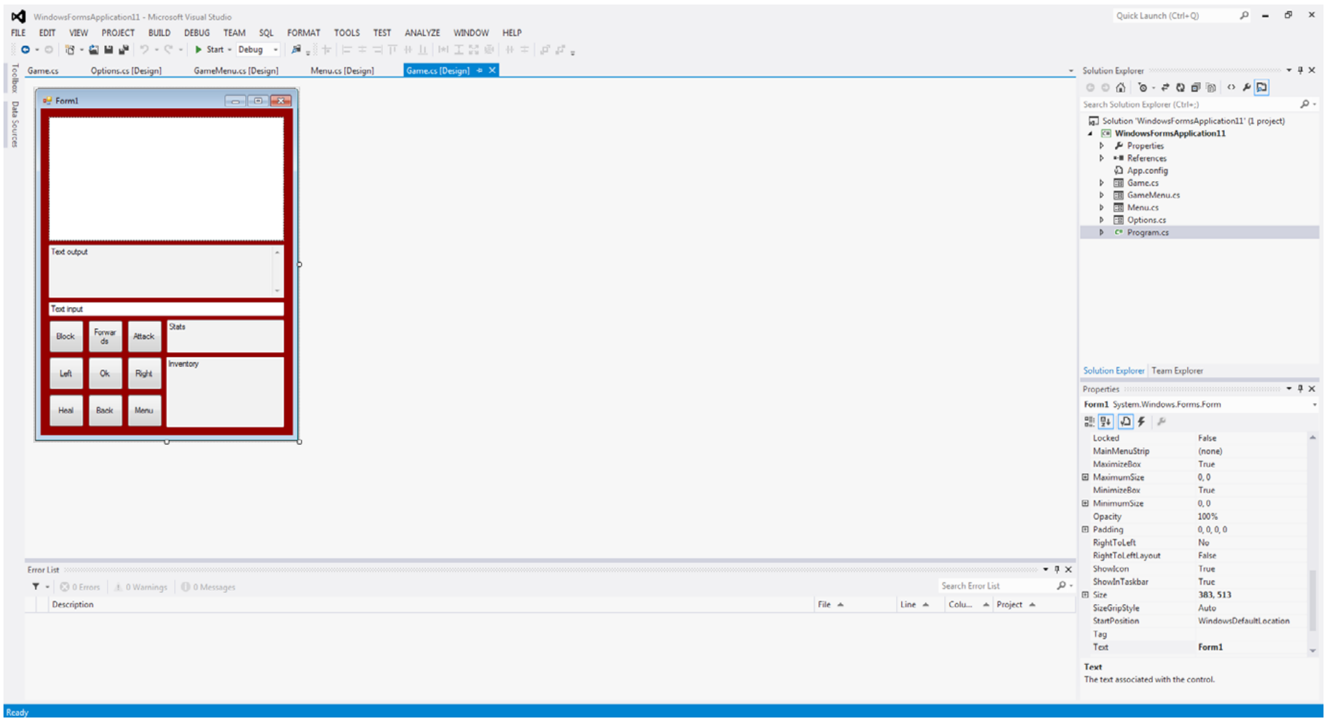


Figure 1: Image of the base room

Using the base shown above I created an image for each room.

Now that I have each room layout and I have already created the interface designs for the game using Visual studio, all I need to do is name the items on the interface and start coding the game.


WindowsFormsApplication11 - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM SQL FORMAT TOOLS TEST ANALYZE WINDOW HELP
Start Debug
Game.cs Options.cs [Design] GameMenu.cs [Design] Menu.cs [Design] Game.cs [Design]
Form1
Text output
Text input
Block Forwards Attack Stats
Left Ok Right Inventory
Heal Back Menu
Solution Explorer
Solution 'WindowsFormsApplication11' (1 project)
WindowsFormsApplication11
Properties
References
App.config
Game.cs
GameMenu.cs
Menu.cs
Options.cs
Program.cs
Error List
0 Errors 0 Warnings 0 Messages
Description File Line Colu... Project
Ready

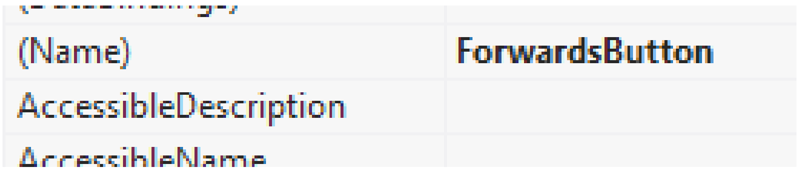
Properties

Form1 System.Windows.Forms.Form

Locked	False
MainMenuStrip	(none)
MaximizeBox	True
MaximumSize	0, 0
MinimizeBox	True
MinimumSize	0, 0
Opacity	100%
Padding	0, 0, 0, 0
RightToLeft	No
RightToLeftLayout	False
ShowIcon	True
ShowInTaskbar	True
Size	383, 513
SizeGripStyle	Auto
StartPosition	WindowsDefaultLocation
Tag	
Text	Form1

Text
The text associated with the control.

To start off I have named each item on the interface with easily identifiable names. For example, movement buttons are simply named the direction they are associated with as shown by the image below of the properties for the forwards button. I did this for each screen. This will make it easier to refer back to each module of the layouts when I am coding.

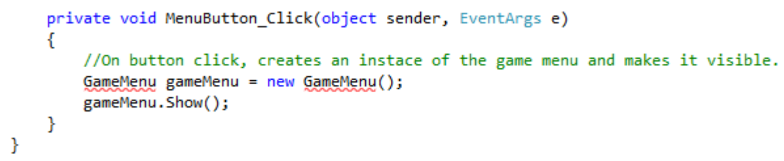


(Name)	ForwardsButton
AccessibleDescription	
AccessibleName	

I decided that it would be best to next work on getting the menu flow working before starting work on the game code.

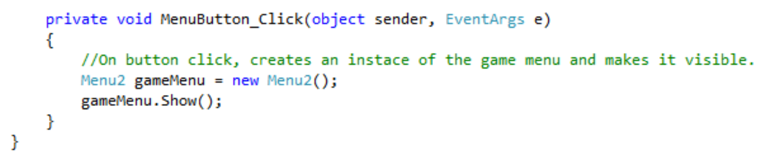
Below is an image of the code that I am using to create instances of forms. I ran into a problem where visual studio was not recognising GameMenu as a class. I realised that this was because the name of the class was not the same as the name of the form. I then changed GameMenu to Menu2 which worked.

Broken



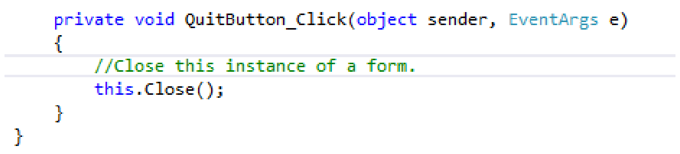
```
        private void MenuButton_Click(object sender, EventArgs e)
        {
            //On button click, creates an instace of the game menu and makes it visible.
            GameMenu gameMenu = new GameMenu();
            gameMenu.Show();
        }
    }
```

Fixed



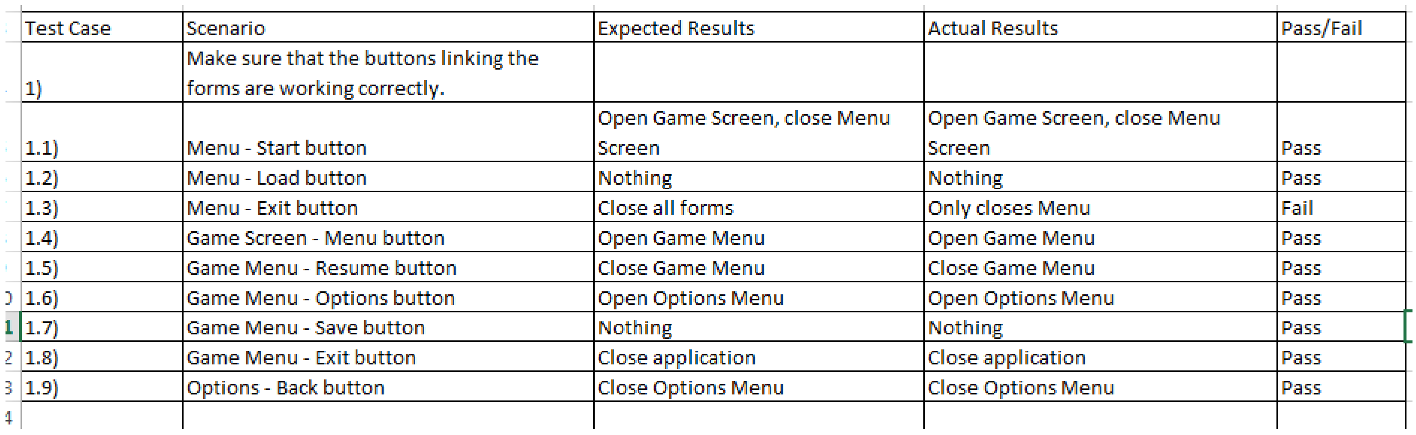
```
        private void MenuButton_Click(object sender, EventArgs e)
        {
            //On button click, creates an instace of the game menu and makes it visible.
            Menu2 gameMenu = new Menu2();
            gameMenu.Show();
        }
    }
```

This simple piece of code is what will be used to close an instance of a form.



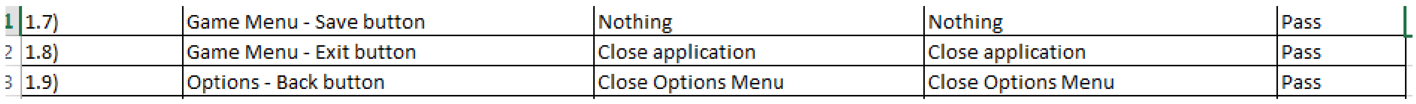
```
        private void QuitButton_Click(object sender, EventArgs e)
        {
            //Close this instance of a form.
            this.Close();
        }
    }
```

After writing the code linking each form I decide to run a test case to document the results of the current code. I created a table in excel to record the results of my tests.



Test Case	Scenario	Expected Results	Actual Results	Pass/Fail
1)	Make sure that the buttons linking the forms are working correctly.			
1.1)	Menu - Start button	Open Game Screen, close Menu Screen	Open Game Screen, close Menu Screen	Pass
1.2)	Menu - Load button	Nothing	Nothing	Pass
1.3)	Menu - Exit button	Close all forms	Only closes Menu	Fail
1.4)	Game Screen - Menu button	Open Game Menu	Open Game Menu	Pass
1.5)	Game Menu - Resume button	Close Game Menu	Close Game Menu	Pass
1.6)	Game Menu - Options button	Open Options Menu	Open Options Menu	Pass
1.7)	Game Menu - Save button	Nothing	Nothing	Pass
1.8)	Game Menu - Exit button	Close application	Close application	Pass
1.9)	Options - Back button	Close Options Menu	Close Options Menu	Pass

From the results I can tell that for the most part I am receiving the correct results however there are a couple of slight problems with some forms not closing when they should.



1.7)	Game Menu - Save button	Nothing	Nothing	Pass
1.8)	Game Menu - Exit button	Close application	Close application	Pass
1.9)	Options - Back button	Close Options Menu	Close Options Menu	Pass

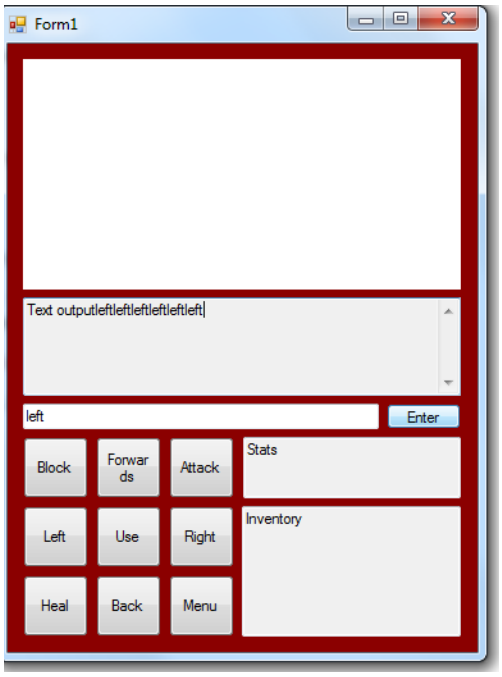
To fix the problem with the exit button on the menu I replaced "This.close" with "Application.Exit". This now shuts down the entire application rather than just the menu form.

The game

To start development of the actual game I am going to create a basic input/output to make sure that the system fulfils the requirements listed in my analysis will work.

I started by creating a method that is called when the enter button is pressed and takes the text currently pressent in the input box and runs an if statement based on the value. For example the user could input "left" which ingame would make the player move to the left however here I have the output box show "left" to show that the system is working.

```
    private void EnterText_Click(object sender, EventArgs e)
    {
        string input = "";
        input = TextInput.Text;

        if (input == "left")
        {
            TextOutput.Text += input;
        }
    }
}
```

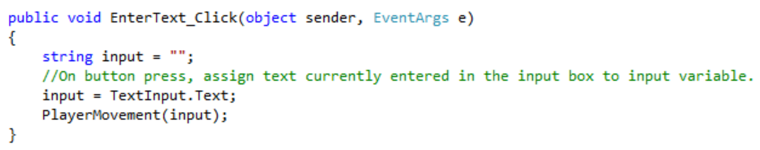


Text outputleftleftleftleftleftleft

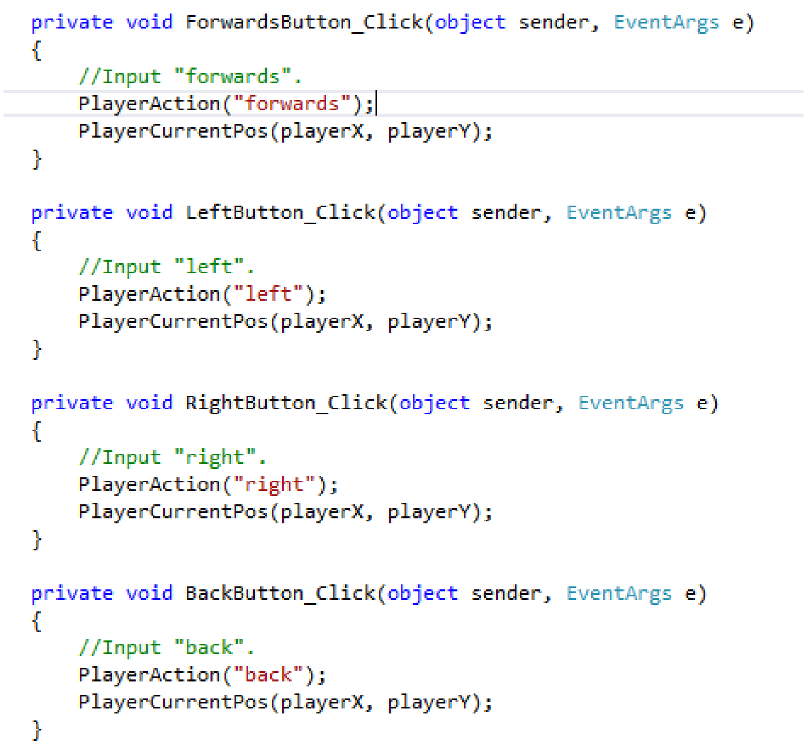
left

This is the snippet of code that I used to test the input/output of my system. From the result it seems as though it works although I would like the text to be created on another line. For now though I will leave it as it is and move on to getting player movement working.

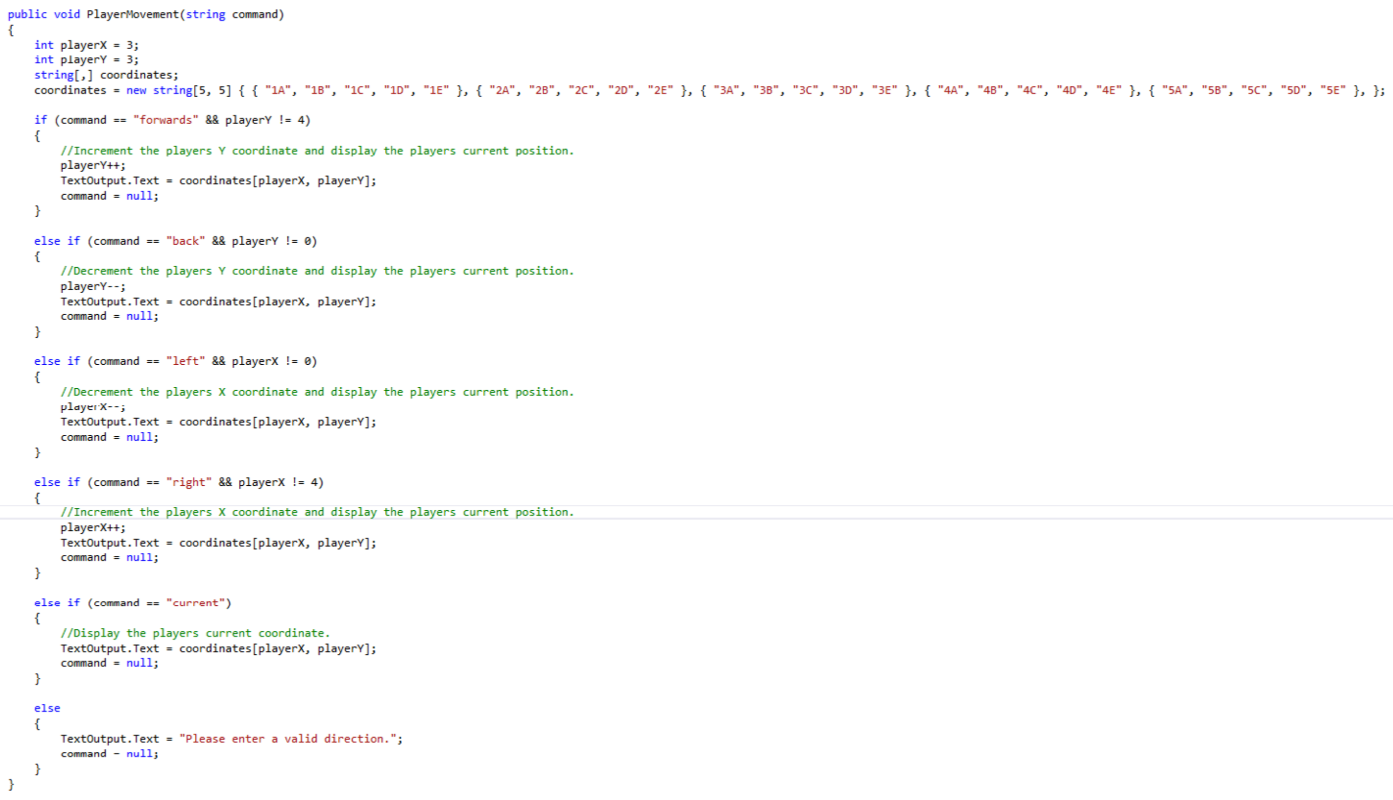
The way I plan to handle player movement is by using an array to store coordinates which point to a specific room. Moving in a direction simply changes the current coordinate.



```
public void EnterText_Click(object sender, EventArgs e)
{
    string input = "";
    //On button press, assign text currently entered in the input box to input variable.
    input = TextInput.Text;
    PlayerMovement(input);
}
```



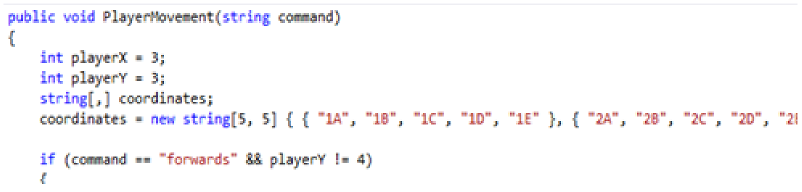
```
private void ForwardsButton_Click(object sender, EventArgs e)
{
    //Input "forwards".
    PlayerAction("forwards");
    PlayerCurrentPos(playerX, playerY);
}

private void LeftButton_Click(object sender, EventArgs e)
{
    //Input "left".
    PlayerAction("left");
    PlayerCurrentPos(playerX, playerY);
}

private void RightButton_Click(object sender, EventArgs e)
{
    //Input "right".
    PlayerAction("right");
    PlayerCurrentPos(playerX, playerY);
}

private void BackButton_Click(object sender, EventArgs e)
{
    //Input "back".
    PlayerAction("back");
    PlayerCurrentPos(playerX, playerY);
}
```



```
public void PlayerMovement(string command)
{
    int playerX = 3;
    int playerY = 3;
    string[,] coordinates;
    coordinates = new string[5, 5] { { "1A", "1B", "1C", "1D", "1E" }, { "2A", "2B", "2C", "2D", "2E" }, { "3A", "3B", "3C", "3D", "3E" }, { "4A", "4B", "4C", "4D", "4E" }, { "5A", "5B", "5C", "5D", "5E" }, };

    if (command == "forwards" && playerY != 4)
    {
        //Increment the players Y coordinate and display the players current position.
        playerY++;
        TextOutput.Text = coordinates[playerX, playerY];
        command = null;
    }

    else if (command == "back" && playerY != 0)
    {
        //Decrement the players Y coordinate and display the players current position.
        playerY--;
        TextOutput.Text = coordinates[playerX, playerY];
        command = null;
    }

    else if (command == "left" && playerX != 0)
    {
        //Decrement the players X coordinate and display the players current position.
        playerX--;
        TextOutput.Text = coordinates[playerX, playerY];
        command = null;
    }

    else if (command == "right" && playerX != 4)
    {
        //Increment the players X coordinate and display the players current position.
        playerX++;
        TextOutput.Text = coordinates[playerX, playerY];
        command = null;
    }

    else if (command == "current")
    {
        //Display the players current coordinate.
        TextOutput.Text = coordinates[playerX, playerY];
        command = null;
    }

    else
    {
        TextOutput.Text = "Please enter a valid direction.";
        command = null;
    }
}
```

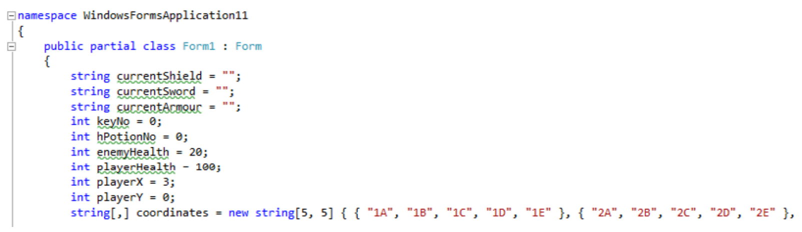
Here is the first version of the code for the players' movement. I created a method called PlayerMovement and put IF statements in it for each direction and one for the displaying the current coordinate. The IF statements run based on the input of the player. If the player inputs "forwards", the players Y coordinate is then incremented and the now current coordinate is displayed in the output box. The method is called when either a button is pressed or a command is entered in the input box. However the problem with the current code is that the coordinates do not inc/decrement and instead each directional input just displays a single coordinate.

I found out that the reason for this problem is that I had declared my variables within the PlayerMovement method resulting in them being reset each time the method was run.



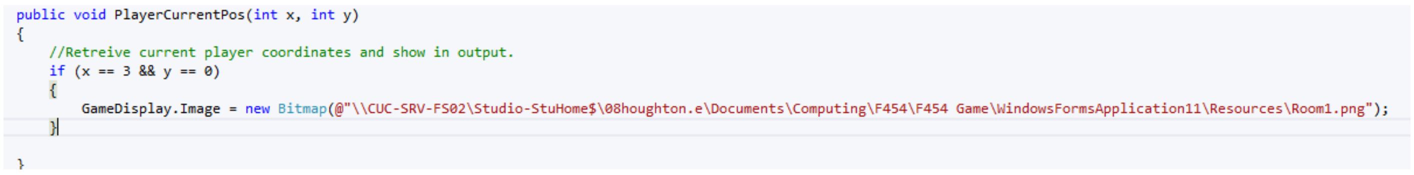
```
public void PlayerMovement(string command)
{
    int playerX = 3;
    int playerY = 3;
    string[,] coordinates;
    coordinates = new string[5, 5] { { "1A", "1B", "1C", "1D", "1E" }, { "2A", "2B", "2C", "2D", "2

    if (command == "forwards" && playerY != 4)
    {
```

To solve this I transferred the variables outside of the method. The coordinates system then produced the desired results. I also decided to rename the PlayerMovement method to PlayerAction as it would be handling actions as well as movement.



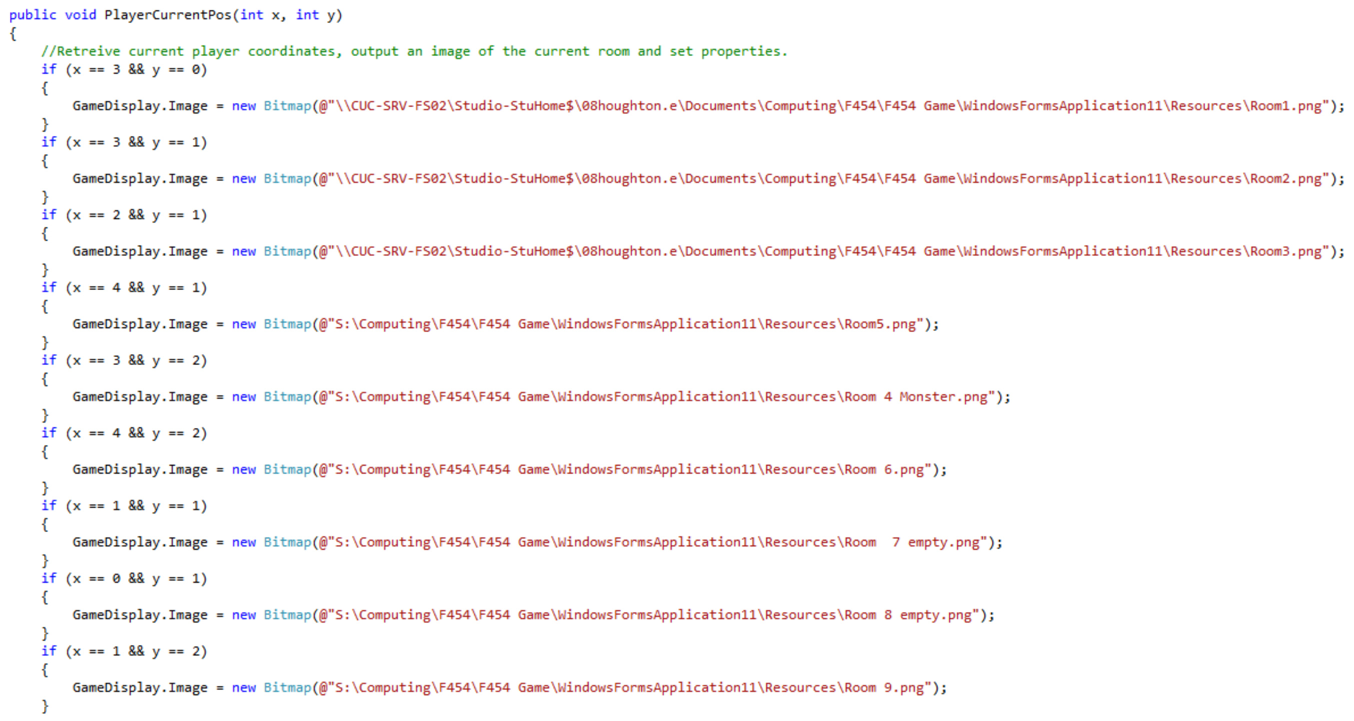
```
namespace WindowsFormsApplication11
{
    public partial class Form1 : Form
    {
        string currentShield = "";
        string currentSword = "";
        string currentArmour = "";
        int keyNo = 0;
        int hPotionNo = 0;
        int enemyHealth = 20;
        int playerHealth = 100;
        int playerX = 3;
        int playerY = 0;
        string[,] coordinates = new string[5, 5] { { "1A", "1B", "1C", "1D", "1E" }, { "2A", "2B", "2C", "2D", "2E" },
```

I now want to have images appear based on the coordinates of the play



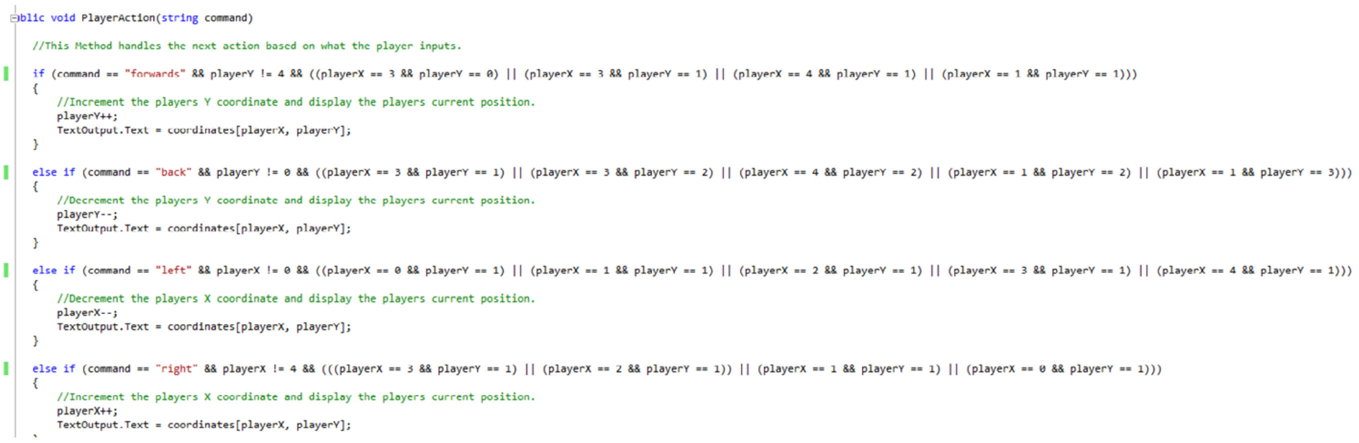
```
public void PlayerCurrentPos(int x, int y)
{
    //Retreive current player coordinates and show in output.
    if (x == 3 && y == 0)
    {
        GameDisplay.Image = new Bitmap(@"\\CUC-SRV-FS02\Studio-StuHome$\08houghton.e\Documents\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room1.png");
    }

}
```

I produced a method called PlayerCurrentPos that contains IF statements that will run based on the current coordinate of the player. I have used this to display a bitmap image of the room the player is currently in however this method will also later be used to set other properties for the rooms. The code worked as I expected it to so I repeated the code for each room.



```
public void PlayerCurrentPos(int x, int y)
{
    //Retreive current player coordinates, output an image of the current room and set properties.
    if (x == 3 && y == 0)
    {
        GameDisplay.Image = new Bitmap(@"\\CUC-SRV-FS02\Studio-StuHome$\08houghton.e\Documents\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room1.png");
    }
    if (x == 3 && y == 1)
    {
        GameDisplay.Image = new Bitmap(@"\\CUC-SRV-FS02\Studio-StuHome$\08houghton.e\Documents\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room2.png");
    }
    if (x == 2 && y == 1)
    {
        GameDisplay.Image = new Bitmap(@"\\CUC-SRV-FS02\Studio-StuHome$\08houghton.e\Documents\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room3.png");
    }
    if (x == 4 && y == 1)
    {
        GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room5.png");
    }
    if (x == 3 && y == 2)
    {
        GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room 4 Monster.png");
    }
    if (x == 4 && y == 2)
    {
        GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room 6.png");
    }
    if (x == 1 && y == 1)
    {
        GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room  7 empty.png");
    }
    if (x == 0 && y == 1)
    {
        GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room 8 empty.png");
    }
    if (x == 1 && y == 2)
    {
        GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room 9.png");
    }
```

The game currently is working as it should however there are no boundaries to restrict the player from walking through walls and entering coordinates that will remain empty and unused. To counteract this I will have to add restrictions to movement between rooms in the PlayerAction method.



```
blic void PlayerAction(string command)

    //This Method handles the next action based on what the player inputs.

    if (command == "forwards" && playerY != 4 && ((playerX == 3 && playerY == 0) || (playerX == 3 && playerY == 1) || (playerX == 4 && playerY == 1) || (playerX == 1 && playerY == 1)))
    {
        //Increment the players Y coordinate and display the players current position.
        playerY++;
        TextOutput.Text = coordinates[playerX, playerY];
    }

    else if (command == "back" && playerY != 0 && ((playerX == 3 && playerY == 1) || (playerX == 3 && playerY == 2) || (playerX == 4 && playerY == 2) || (playerX == 1 && playerY == 2) || (playerX == 1 && playerY == 3)))
    {
        //Decrement the players Y coordinate and display the players current position.
        playerY--;
        TextOutput.Text = coordinates[playerX, playerY];
    }

    else if (command == "left" && playerX != 0 && ((playerX == 0 && playerY == 1) || (playerX == 1 && playerY == 1) || (playerX == 2 && playerY == 1) || (playerX == 3 && playerY == 1) || (playerX == 4 && playerY == 1)))
    {
        //Decrement the players X coordinate and display the players current position.
        playerX--;
        TextOutput.Text = coordinates[playerX, playerY];
    }

    else if (command == "right" && playerX != 4 && (((playerX == 3 && playerY == 1) || (playerX == 2 && playerY == 1)) || (playerX == 1 && playerY == 1) || (playerX == 0 && playerY == 1)))
    {
        //Increment the players X coordinate and display the players current position.
        playerX++;
        TextOutput.Text = coordinates[playerX, playerY];
```

The image above shows the PlayerAction method after I have added restrictions on which rooms commands can be used in. I didn't run into any serious problems while adding the restrictions however there was a few times where I hadn't included a room or the problem was easily fixable such as missing parentheses.

Now that navigation is completely done I will start working on the combat system.

I pre-emptively created some variables that may prove useful later as shown below.

```
public partial class Form1 : Form
{
    //Whether the player can pass through the door.
    bool riddleDoor1 = false;
    bool passDoor1 = false;

    //The equipment currenly being utilised by the player. these act as multiplyers for their respective uses.
    int currentShieldValue = 1; //Block chance.
    int currentSwordValue = 1; //Sword damage.
    int currentArmourValue = 1; //Damage reduction.

    //Number of items in the players inventory.
    int keyNo = 0;
    int hPotionNo = 0;
    string shield = "Rotten buckler";
    string sword = "Wooden club";
    string armour = "Old rags";

    //Various player and enemy properties.
    bool enemy3Alive = true;
    bool enemy2Alive = true;
    bool enemy1Alive = true;
    int enemy3Health = 60;
    int enemy2Health = 30;
    int enemy1Health = 20;
    int playerHealth = 100;
    Random chance = new Random();
```

I am going to create a basic random damage, turn based combat system to start out. I will then add in the other features such as healing and blocking.

During my creation of the combat code I ran into a problem. I had to call have

Issues I had in development

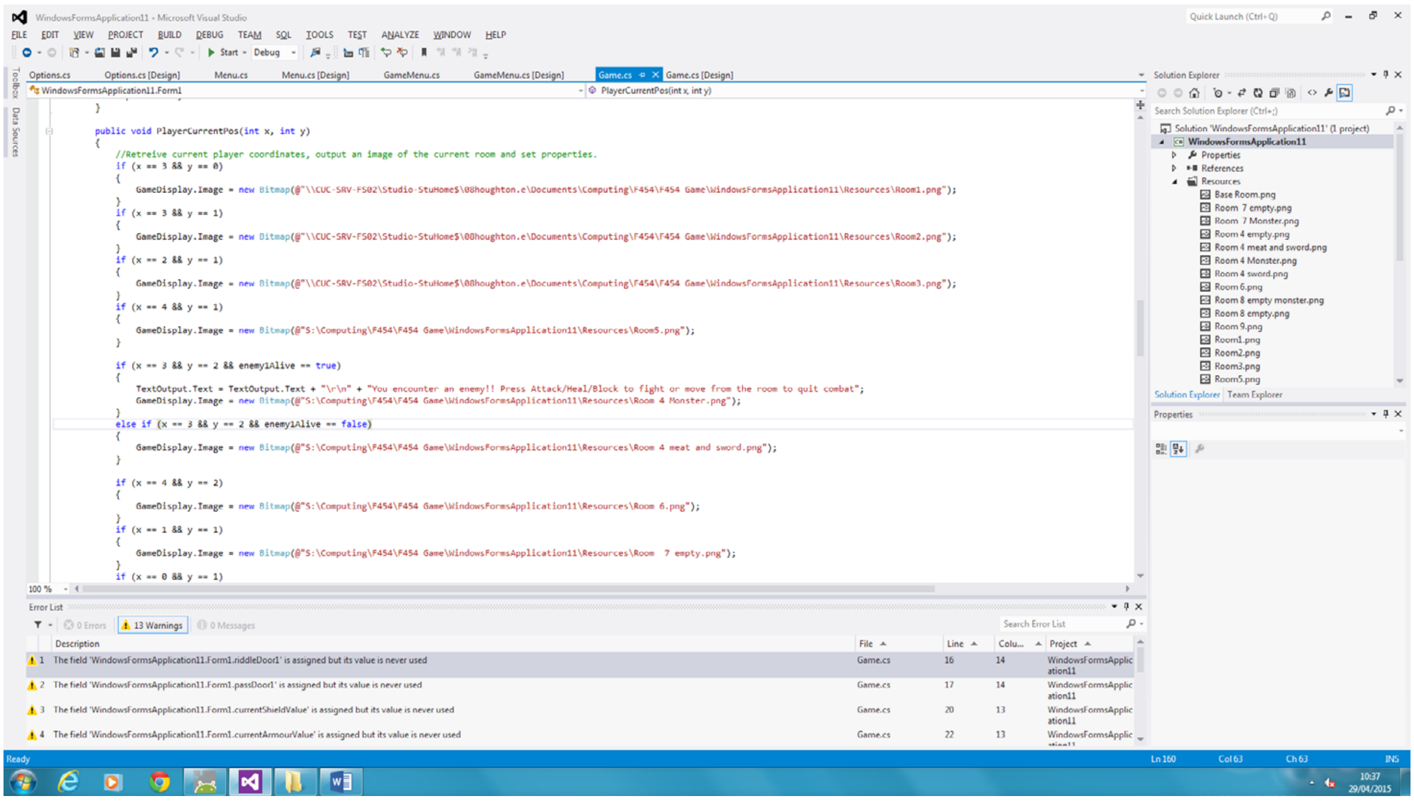
- Calling Method
  - Issues with how to get the actions in through use of buttons
  - Decided to pass strings to Combat Method when buttons clicked
- Clicking Attack button
  - Clicking attack set a string Action to "attack" and passed into combat method
- Getting stuck in While Loop
- Kept calling/resetting monster health
- Extracted each "action" out of combat method into individual function that was called from the "Combat" method
- Visibility of variables
  - Player health visible
  - Had to get monster health in based on room
- Selecting correct monster health
  - Added IF Statements to cover this
- Created new Attack Method
- Created new Block method
- Created new Heal method

While creating the combat system I ran into a few problems. At first I had issues trying to get the actions to work through the use of buttons. I couldn't call the method for when the "Enter" was clicked into the CombatSystem method. I fixed this problem by passing strings containing the action through the CombatSystem method using three other methods that are called when any of the three combat buttons are clicked. The action and location of the player are compared to IF statements contained within the method to produce the final action. The actions contained within the combat method were extracted out into their own methods for this to work.

The original combat method contained a WHILE loop that reset the variable representing the health of the enemy with each loop. By removing the loop I fixed this problem.

In order for the program to recognise which enemy's health to use I used IF statements that compared the players current coordinates to know if an enemy was present and if so, which one.

I added a message that would be outputted to the text output box once a player entered room 3, 2 telling the player what to do when encountering an enemy.


WindowsFormsApplication11 - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM SQL TOOLS TEST ANALYZE WINDOW HELP
Start Debug
Options.cs Options.cs [Design] Menu.cs Menu.cs [Design] GameMenu.cs GameMenu.cs [Design] Game.cs Game.cs [Design]
WindowsFormsApplication11.Form1
PlayerCurrentPos(int x, int y)
Solution Explorer
Solution 'WindowsFormsApplication11' (1 project)
WindowsFormsApplication11
Properties
References
Resources
Base Room.png
Room 7 empty.png
Room 7 Monster.png
Room 4 empty.png
Room 4 meat and sword.png
Room 4 Monster.png
Room 4 sword.png
Room 6.png
Room 8 empty monster.png
Room 8 empty.png
Room 9.png
Room1.png
Room2.png
Room3.png
Room5.png
Properties
Ready Ln 160 Col 63 Ch 63 INS
10:37 29/04/2015

```
        }

        public void PlayerCurrentPos(int x, int y)
        {
            //Retreive current player coordinates, output an image of the current room and set properties.
            if (x == 3 && y == 0)
            {
                GameDisplay.Image = new Bitmap(@"\\CUC-SRV-FS02\Studio-StuHome$\08houghton.e\Documents\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room1.png");
            }
            if (x == 3 && y == 1)
            {
                GameDisplay.Image = new Bitmap(@"\\CUC-SRV-FS02\Studio-StuHome$\08houghton.e\Documents\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room2.png");
            }
            if (x == 2 && y == 1)
            {
                GameDisplay.Image = new Bitmap(@"\\CUC-SRV-FS02\Studio-StuHome$\08houghton.e\Documents\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room3.png");
            }
            if (x == 4 && y == 1)
            {
                GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room5.png");
            }

            if (x == 3 && y == 2 && enemy1Alive == true)
            {
                TextOutput.Text = TextOutput.Text + "\r\n" + "You encounter an enemy!! Press Attack/Heal/Block to fight or move from the room to quit combat";
                GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room 4 Monster.png");
            }
            else if (x == 3 && y == 2 && enemy1Alive == false)
            {
                GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room 4 meat and sword.png");
            }

            if (x == 4 && y == 2)
            {
                GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room 6.png");
            }
            if (x == 1 && y == 1)
            {
                GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room  7 empty.png");
            }
            if (x == 0 && y == 1)
```

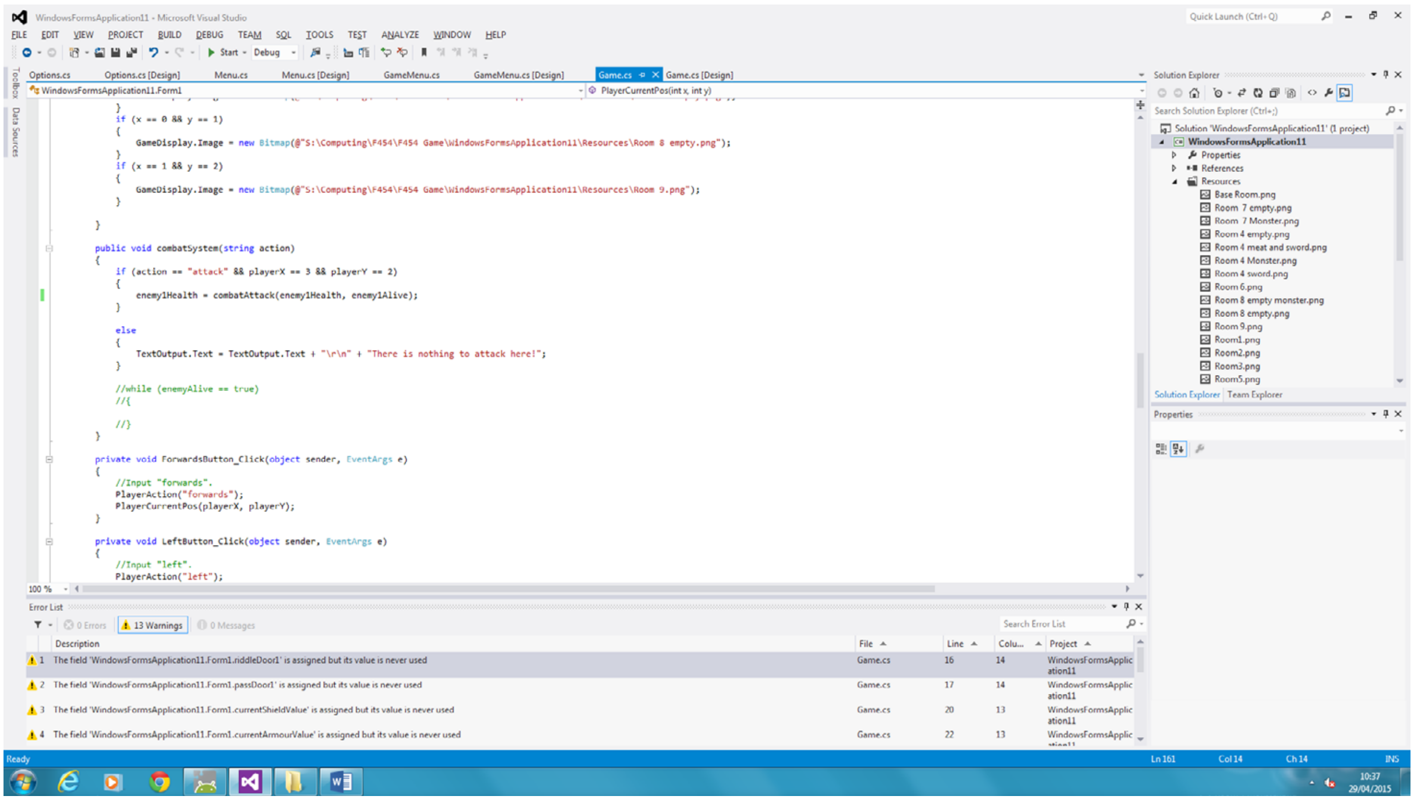
Error List

0 Errors | 13 Warnings | 0 Messages

	Description	File	Line	Colu...	Project
1	The field 'WindowsFormsApplication11.Form1.riddleDoor1' is assigned but its value is never used	Game.cs	16	14	WindowsFormsApplication11
2	The field 'WindowsFormsApplication11.Form1.passDoor1' is assigned but its value is never used	Game.cs	17	14	WindowsFormsApplication11
3	The field 'WindowsFormsApplication11.Form1.currentShieldValue' is assigned but its value is never used	Game.cs	20	13	WindowsFormsApplication11
4	The field 'WindowsFormsApplication11.Form1.currentArmourValue' is assigned but its value is never used	Game.cs	22	13	WindowsFormsApplication11

Figure 2: Adding descriptions for user on attacking monster

I removed the WHILE loop as It was not necessary

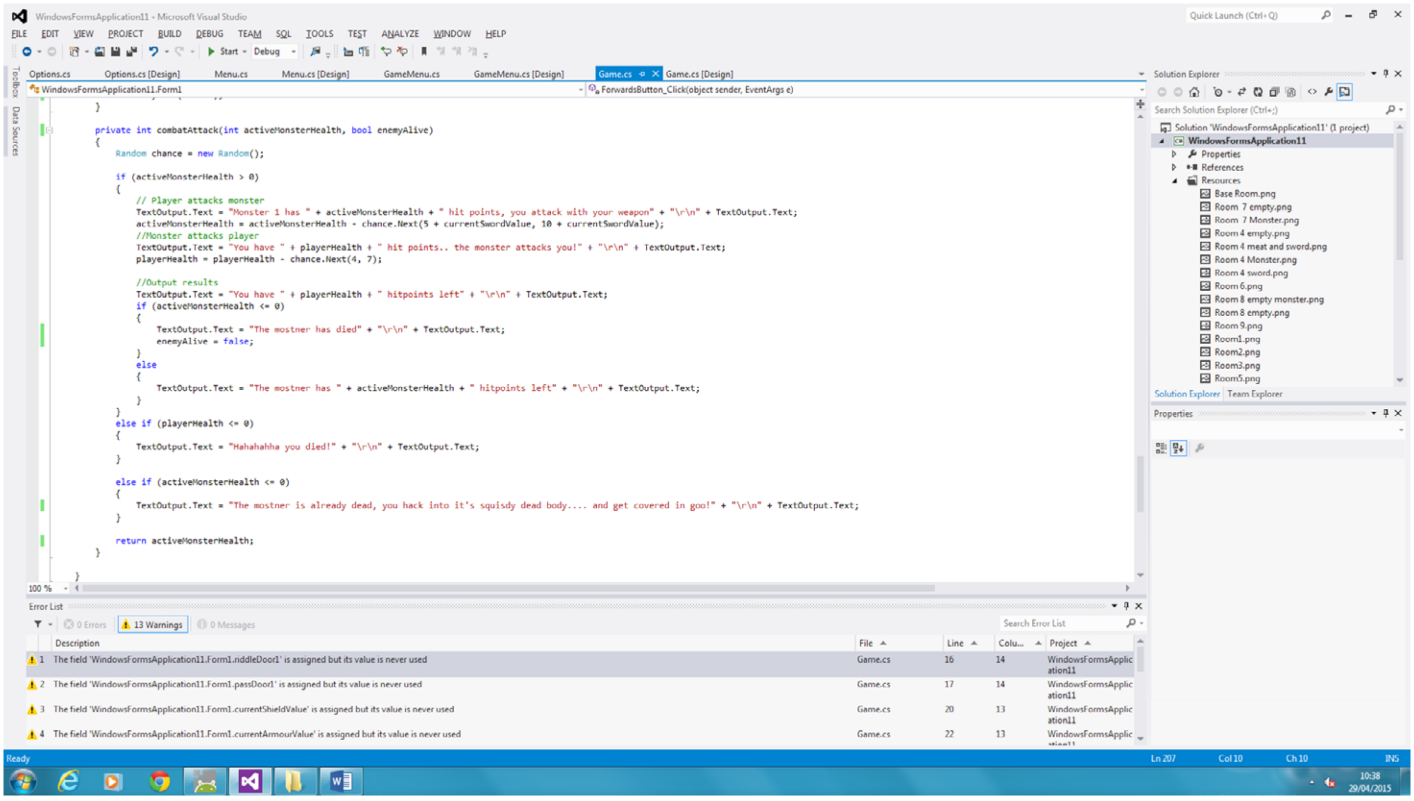


```
            }
            if (x == 0 && y == 1)
            {
                GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room 8 empty.png");
            }
            if (x == 1 && y == 2)
            {
                GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room 9.png");
            }

        }

        public void combatSystem(string action)
        {
            if (action == "attack" && playerX == 3 && playerY == 2)
            {
                enemy1Health = combatAttack(enemy1Health, enemy1Alive);
            }

            else
            {
                TextOutput.Text = TextOutput.Text + "\r\n" + "There is nothing to attack here!";
            }

            //while (enemyAlive == true)
            //{

            //}
        }

        private void ForwardsButton_Click(object sender, EventArgs e)
        {
            //Input "forwards".
            PlayerAction("forwards");
            PlayerCurrentPos(playerX, playerY);
        }

        private void LeftButton_Click(object sender, EventArgs e)
        {
            //Input "left".
            PlayerAction("left");
```

```
Error List
0 Errors   13 Warnings   0 Messages
   Description                                                                                   File      Line  Colu...  Project
1  The field 'WindowsFormsApplication11.Form1.riddleDoor1' is assigned but its value is never used        Game.cs   16    14       WindowsFormsApplication11
2  The field 'WindowsFormsApplication11.Form1.passDoor1' is assigned but its value is never used          Game.cs   17    14       WindowsFormsApplication11
3  The field 'WindowsFormsApplication11.Form1.currentShieldValue' is assigned but its value is never used  Game.cs   20    13       WindowsFormsApplication11
4  The field 'WindowsFormsApplication11.Form1.currentArmourValue' is assigned but its value is never used  Game.cs   22    13       WindowsFormsApplic
```

Figure 3: removal of While loop and adding new IF statement

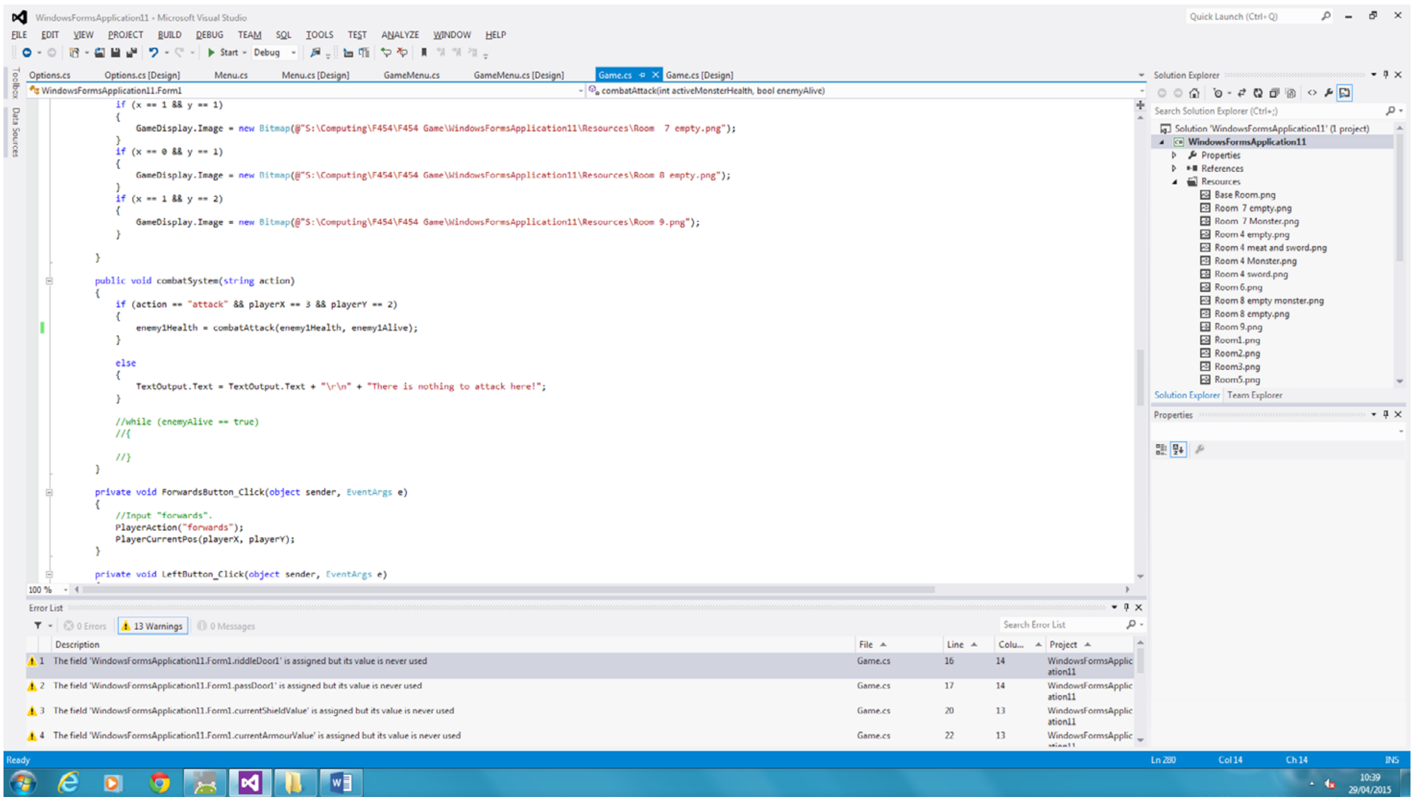
I created a new method called CombatAttack which would be called if the player entered "attack" as their action in combat. It takes the current rooms enemy's health and subtracts a random amount between two numbers and adds the currently equipped swords damage to each number. The same is done to the players health and the process is repeated until either the player or enemy dies or the player leaves the room.



```
        }

        private int combatAttack(int activeMonsterHealth, bool enemyAlive)
        {
            Random chance = new Random();

            if (activeMonsterHealth > 0)
            {
                // Player attacks monster
                TextOutput.Text = "Monster 1 has " + activeMonsterHealth + " hit points, you attack with your weapon" + "\r\n" + TextOutput.Text;
                activeMonsterHealth = activeMonsterHealth - chance.Next(5 + currentSwordValue, 10 + currentSwordValue);
                //Monster attacks player
                TextOutput.Text = "You have " + playerHealth + " hit points.. the monster attacks you!" + "\r\n" + TextOutput.Text;
                playerHealth = playerHealth - chance.Next(4, 7);

                //Output results
                TextOutput.Text = "You have " + playerHealth + " hitpoints left" + "\r\n" + TextOutput.Text;
                if (activeMonsterHealth <= 0)
                {
                    TextOutput.Text = "The mostner has died" + "\r\n" + TextOutput.Text;
                    enemyAlive = false;
                }
                else
                {
                    TextOutput.Text = "The mostner has " + activeMonsterHealth + " hitpoints left" + "\r\n" + TextOutput.Text;
                }
            }
            else if (playerHealth <= 0)
            {
                TextOutput.Text = "Hahahahha you died!" + "\r\n" + TextOutput.Text;
            }

            else if (activeMonsterHealth <= 0)
            {
                TextOutput.Text = "The mostner is already dead, you hack into it's squisdy dead body.... and get covered in goo!" + "\r\n" + TextOutput.Text;
            }

            return activeMonsterHealth;
        }

    }
```

```
Error List
0 Errors   13 Warnings   0 Messages
   Description                                                                                   File      Line  Colu...  Project
1  The field 'WindowsFormsApplication11.Form1.riddleDoor1' is assigned but its value is never used        Game.cs   16    14       WindowsFormsApplication11
2  The field 'WindowsFormsApplication11.Form1.passDoor1' is assigned but its value is never used          Game.cs   17    14       WindowsFormsApplication11
3  The field 'WindowsFormsApplication11.Form1.currentShieldValue' is assigned but its value is never used  Game.cs   20    13       WindowsFormsApplication11
4  The field 'WindowsFormsApplication11.Form1.currentArmourValue' is assigned but its value is never used  Game.cs   22    13       WindowsFormsApplic
```

Figure 4: Creation of new Attack Method, called from the **CombatAttack** method.

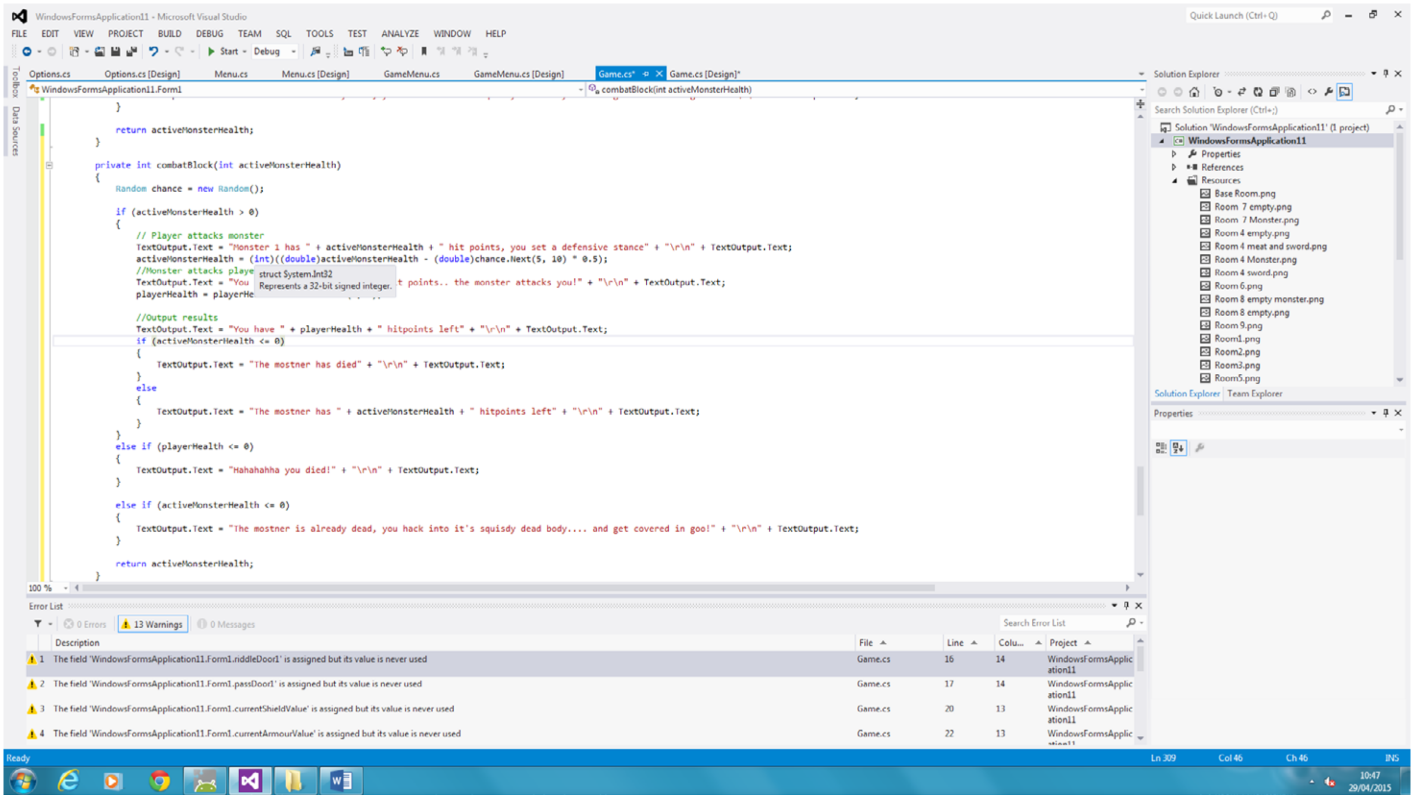


```
            if (x == 1 && y == 1)
            {
                GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room  7 empty.png");
            }
            if (x == 0 && y == 1)
            {
                GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room 8 empty.png");
            }
            if (x == 1 && y == 2)
            {
                GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room 9.png");
            }

        }

        public void combatSystem(string action)
        {
            if (action == "attack" && playerX == 3 && playerY == 2)
            {
                enemy1Health = combatAttack(enemy1Health, enemy1Alive);
            }

            else
            {
                TextOutput.Text = TextOutput.Text + "\r\n" + "There is nothing to attack here!";
            }

            //while (enemyAlive == true)
            //{

            //}
        }

        private void ForwardsButton_Click(object sender, EventArgs e)
        {
            //Input "forwards".
            PlayerAction("forwards");
            PlayerCurrentPos(playerX, playerY);
        }

        private void LeftButton_Click(object sender, EventArgs e)
```

```
Error List
0 Errors   13 Warnings   0 Messages

   Description                                                                          File     Line  Colu...  Project
1  The field 'WindowsFormsApplication11.Form1.riddleDoor1' is assigned but its value is never used         Game.cs  16    14   WindowsFormsApplication11
2  The field 'WindowsFormsApplication11.Form1.passDoor1' is assigned but its value is never used           Game.cs  17    14   WindowsFormsApplication11
3  The field 'WindowsFormsApplication11.Form1.currentShieldValue' is assigned but its value is never used  Game.cs  20    13   WindowsFormsApplication11
4  The field 'WindowsFormsApplication11.Form1.currentArmourValue' is assigned but its value is never used  Game.cs  22    13   WindowsFormsApplic
```

Figure 5: Removal of WHILE loop and code from Combat Method to new Attack Method.

If the player chooses to block during their turn in combat, their attack damage is halved but they have a 50/50 chance of negating all damage received. Due to the attack damage needing to be multiplied by 0.5 there was the chance that it would result in a decimal number, therefore some values needed to be temporarily converted to doubles and back to integers.

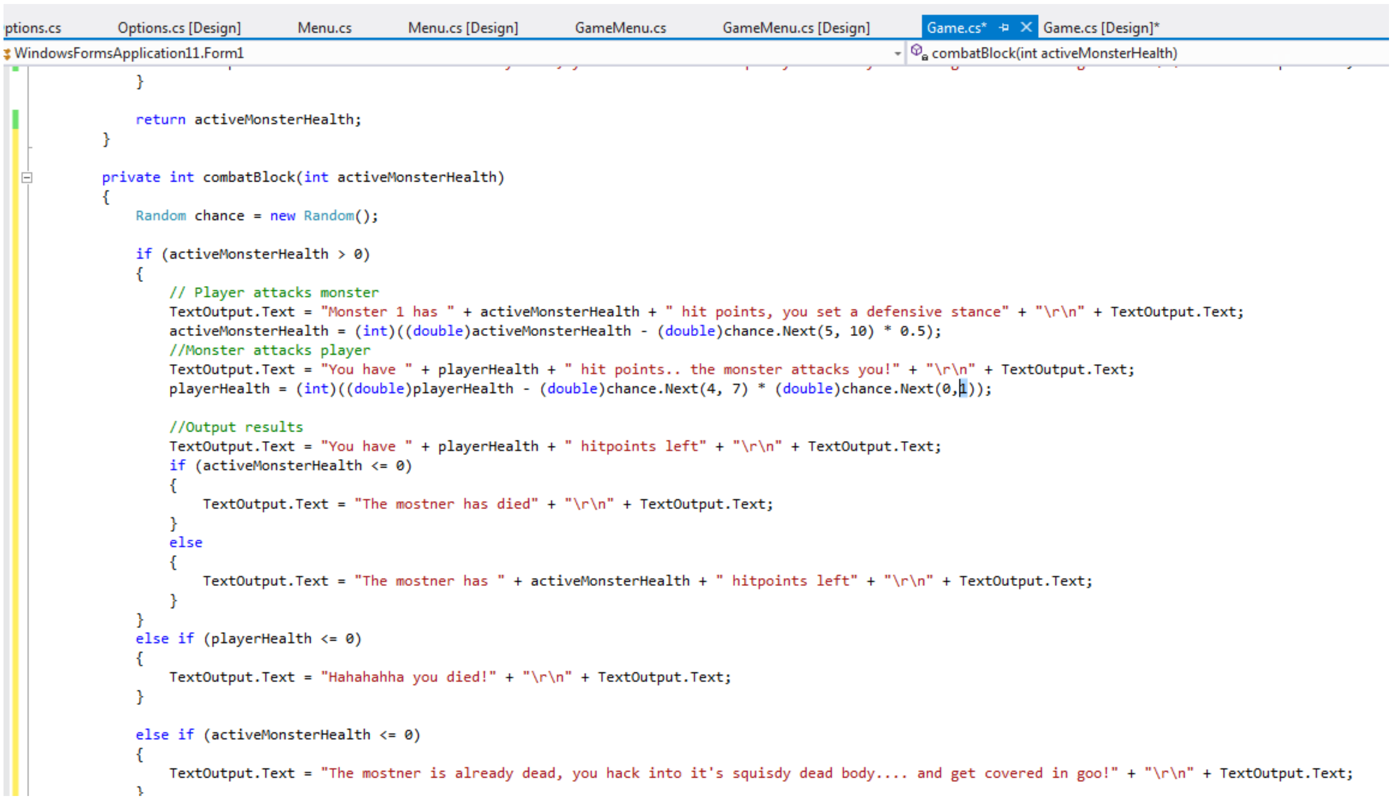


```
            }

            return activeMonsterHealth;
        }

        private int combatBlock(int activeMonsterHealth)
        {
            Random chance = new Random();

            if (activeMonsterHealth > 0)
            {
                // Player attacks monster
                TextOutput.Text = "Monster 1 has " + activeMonsterHealth + " hit points, you set a defensive stance" + "\r\n" + TextOutput.Text;
                activeMonsterHealth = (int)((double)activeMonsterHealth - (double)chance.Next(5, 10) * 0.5);
                //Monster attacks playe
                TextOutput.Text = "You                              t points.. the monster attacks you!" + "\r\n" + TextOutput.Text;
                playerHealth = playerHe

                //Output results
                TextOutput.Text = "You have " + playerHealth + " hitpoints left" + "\r\n" + TextOutput.Text;
                if (activeMonsterHealth <= 0)
                {
                    TextOutput.Text = "The mostner has died" + "\r\n" + TextOutput.Text;
                }
                else
                {
                    TextOutput.Text = "The mostner has " + activeMonsterHealth + " hitpoints left" + "\r\n" + TextOutput.Text;
                }
            }
            else if (playerHealth <= 0)
            {
                TextOutput.Text = "Hahahahha you died!" + "\r\n" + TextOutput.Text;
            }

            else if (activeMonsterHealth <= 0)
            {
                TextOutput.Text = "The mostner is already dead, you hack into it's squisdy dead body.... and get covered in goo!" + "\r\n" + TextOutput.Text;
            }

            return activeMonsterHealth;
        }
```

```
struct System.Int32
Represents a 32-bit signed integer.
```

```
Error List
0 Errors   13 Warnings   0 Messages

   Description                                                                          File     Line  Colu...  Project
1  The field 'WindowsFormsApplication11.Form1.riddleDoor1' is assigned but its value is never used         Game.cs  16    14   WindowsFormsApplication11
2  The field 'WindowsFormsApplication11.Form1.passDoor1' is assigned but its value is never used           Game.cs  17    14   WindowsFormsApplication11
3  The field 'WindowsFormsApplication11.Form1.currentShieldValue' is assigned but its value is never used  Game.cs  20    13   WindowsFormsApplication11
4  The field 'WindowsFormsApplication11.Form1.currentArmourValue' is assigned but its value is never used  Game.cs  22    13   WindowsFormsApplic
```

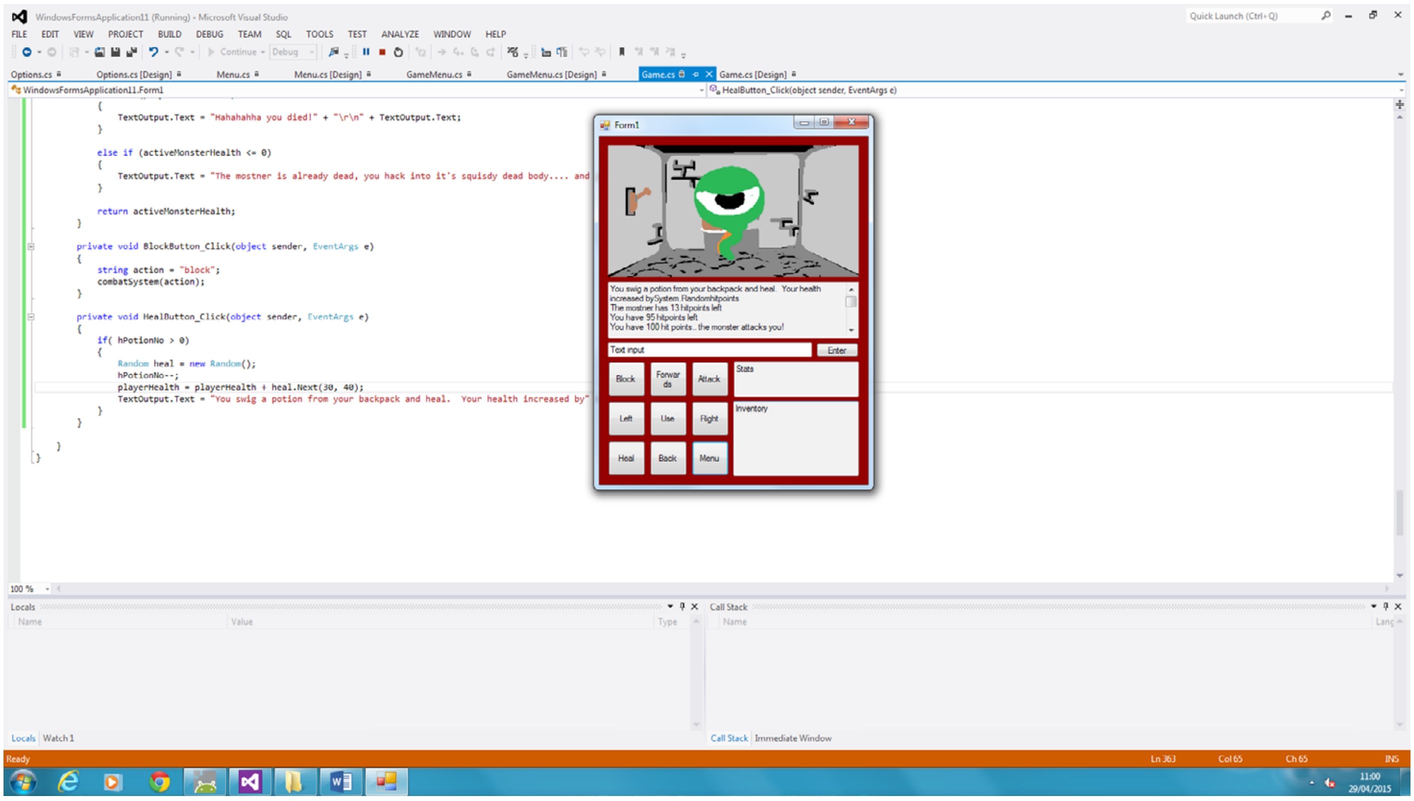
Figure 6: Had to convert to doubles for 0.5 multiplier and then back to Int



```
            }

            return activeMonsterHealth;
        }

        private int combatBlock(int activeMonsterHealth)
        {
            Random chance = new Random();

            if (activeMonsterHealth > 0)
            {
                // Player attacks monster
                TextOutput.Text = "Monster 1 has " + activeMonsterHealth + " hit points, you set a defensive stance" + "\r\n" + TextOutput.Text;
                activeMonsterHealth = (int)((double)activeMonsterHealth - (double)chance.Next(5, 10) * 0.5);
                //Monster attacks player
                TextOutput.Text = "You have " + playerHealth + " hit points.. the monster attacks you!" + "\r\n" + TextOutput.Text;
                playerHealth = (int)((double)playerHealth - (double)chance.Next(4, 7) * (double)chance.Next(0,1));

                //Output results
                TextOutput.Text = "You have " + playerHealth + " hitpoints left" + "\r\n" + TextOutput.Text;
                if (activeMonsterHealth <= 0)
                {
                    TextOutput.Text = "The mostner has died" + "\r\n" + TextOutput.Text;
                }
                else
                {
                    TextOutput.Text = "The mostner has " + activeMonsterHealth + " hitpoints left" + "\r\n" + TextOutput.Text;
                }
            }
            else if (playerHealth <= 0)
            {
                TextOutput.Text = "Hahahahha you died!" + "\r\n" + TextOutput.Text;
            }

            else if (activeMonsterHealth <= 0)
            {
                TextOutput.Text = "The mostner is already dead, you hack into it's squisdy dead body.... and get covered in goo!" + "\r\n" + TextOutput.Text;
```

Figure 7: Added block method that reduced player dmg, but gives player a 50/50 to stop ALL incoming damage.

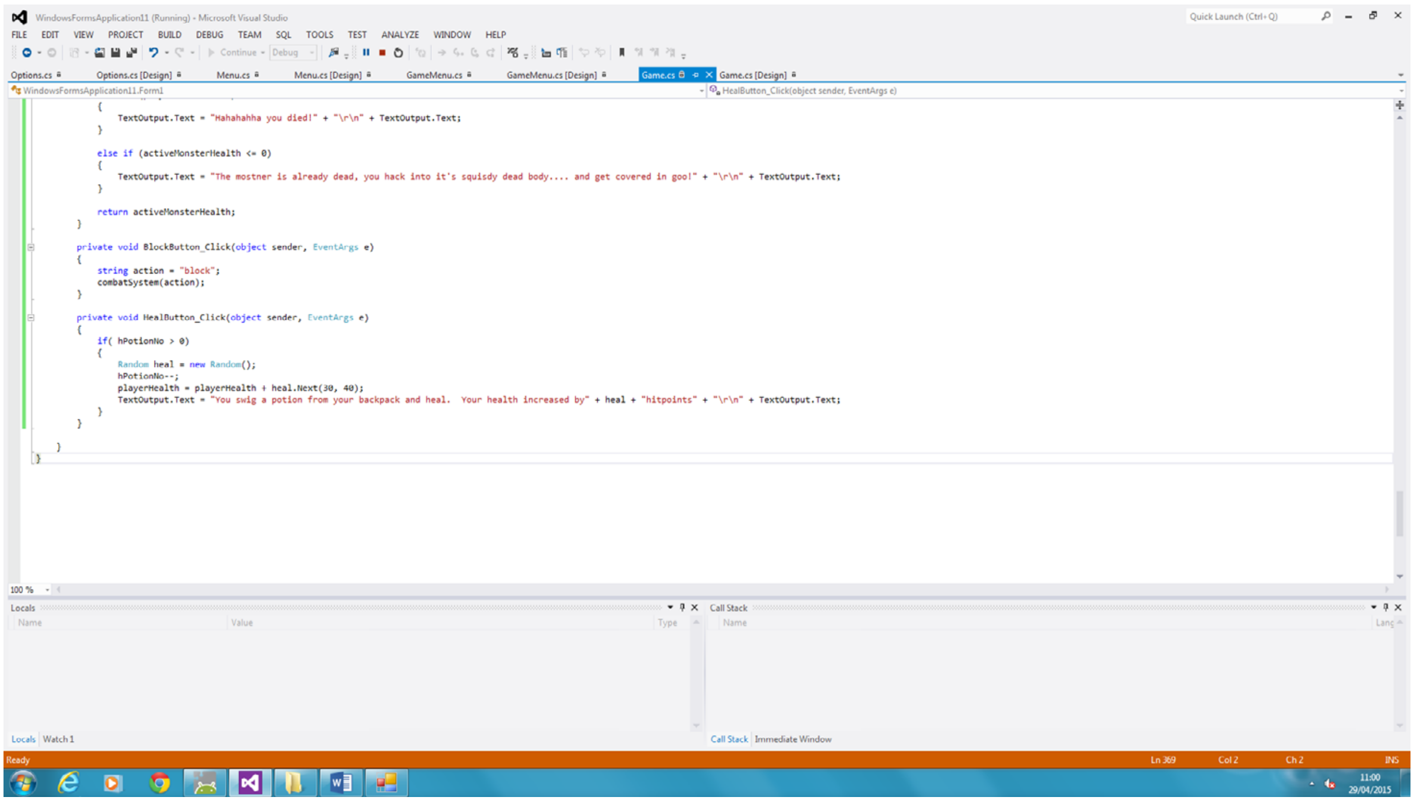
While creating the healing method I ran into the problem that using the random number generator "heal" as a variable resulted in the text output box showing "System.Randomhitpoints" instead of the value that heal produced. I fixed this by assigning the value produced by heal to a variable called "healthBoost" and outputting that in the text output box.



```
            {
                TextOutput.Text = "Hahahahha you died!" + "\r\n" + TextOutput.Text;
            }

            else if (activeMonsterHealth <= 0)
            {
                TextOutput.Text = "The mostner is already dead, you hack into it's squisdy dead body.... and
            }

            return activeMonsterHealth;
        }

        private void BlockButton_Click(object sender, EventArgs e)
        {
            string action = "block";
            combatSystem(action);
        }

        private void HealButton_Click(object sender, EventArgs e)
        {
            if( hPotionNo > 0)
            {
                Random heal = new Random();
                hPotionNo--;
                playerHealth = playerHealth + heal.Next(30, 40);
                TextOutput.Text = "You swig a potion from your backpack and heal.  Your health increased by"
            }
        }

    }
}
```

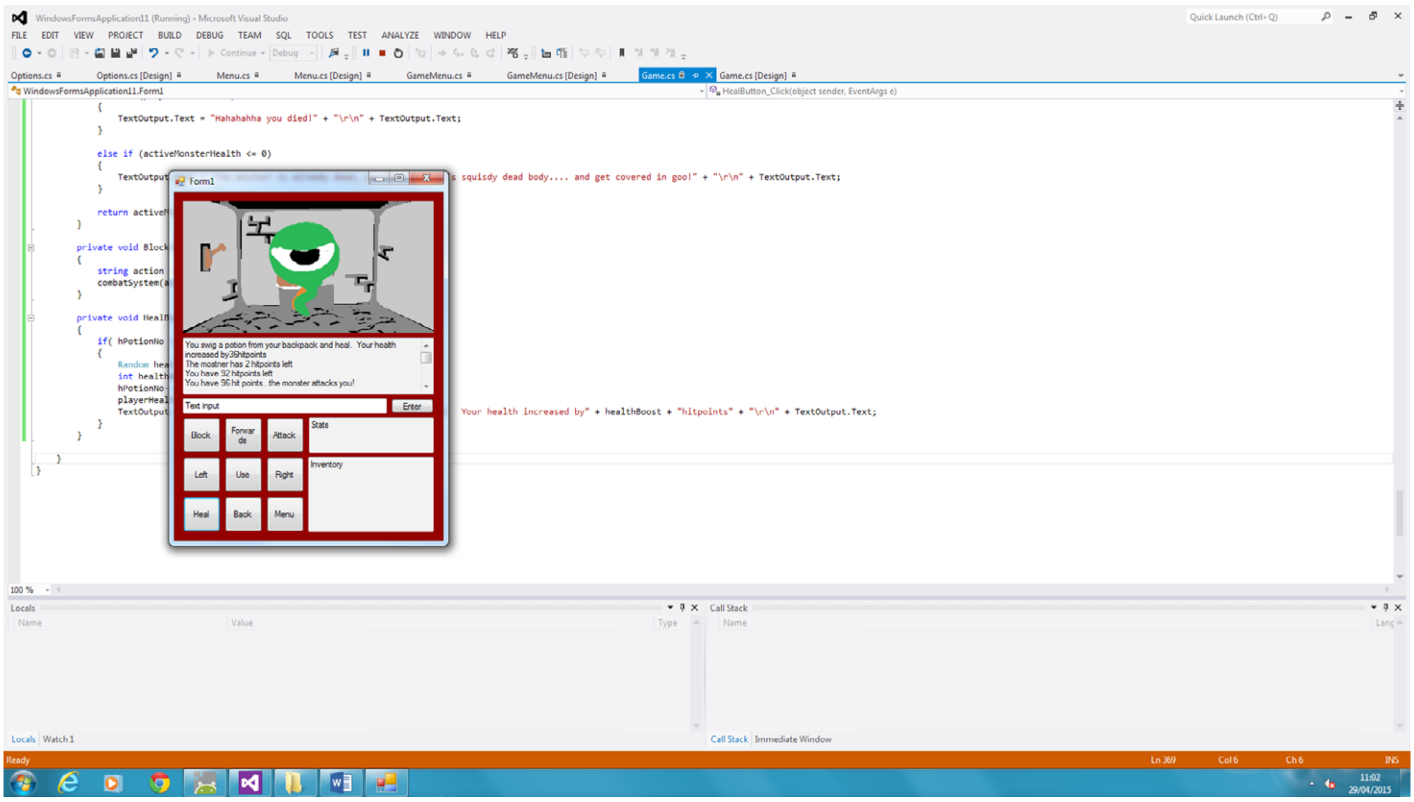
```
You swig a potion from your backpack and heal.  Your health
increased bySystem.Randomhitpoints
The mostner has 13 hitpoints left
You have 95 hitpoints left
You have 100 hit points.. the monster attacks you!
```

Figure 8: Started heal method. Tried to use "heal" as a variable. Failed!



```
            {
                TextOutput.Text = "Hahahahha you died!" + "\r\n" + TextOutput.Text;
            }

            else if (activeMonsterHealth <= 0)
            {
                TextOutput.Text = "The mostner is already dead, you hack into it's squisdy dead body.... and get covered in goo!" + "\r\n" + TextOutput.Text;
            }

            return activeMonsterHealth;
        }

        private void BlockButton_Click(object sender, EventArgs e)
        {
            string action = "block";
            combatSystem(action);
        }

        private void HealButton_Click(object sender, EventArgs e)
        {
            if( hPotionNo > 0)
            {
                Random heal = new Random();
                hPotionNo--;
                playerHealth = playerHealth + heal.Next(30, 40);
                TextOutput.Text = "You swig a potion from your backpack and heal.  Your health increased by" + heal + "hitpoints" + "\r\n" + TextOutput.Text;
            }
        }

    }
}
```

Figure 9: Heal code that is not working.

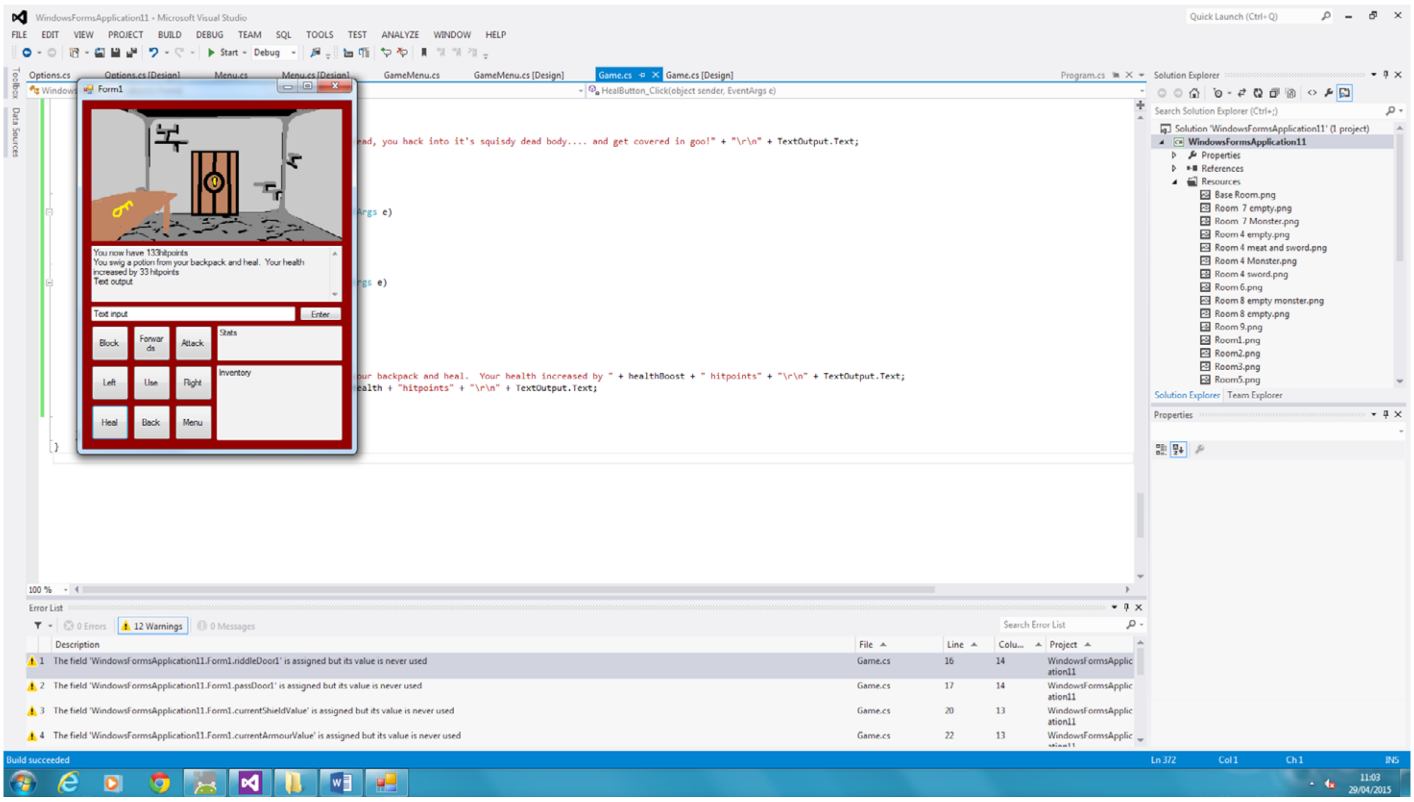


```
            {
                TextOutput.Text = "Hahahahha you died!" + "\r\n" + TextOutput.Text;
            }

            else if (activeMonsterHealth <= 0)
            {
                TextOutput                                   s squisdy dead body.... and get covered in goo!" + "\r\n" + TextOutput.Text;
            }

            return activeM
        }

        private void Block
        {
            string action
            combatSystem(a
        }

        private void HealB
        {
            if( hPotionNo
            {
                Random hea
                int health
                hPotionNo-
                playerHeal
                TextOutput                                   Your health increased by" + healthBoost + "hitpoints" + "\r\n" + TextOutput.Text;
            }
        }

    }
}
```

```
You swig a potion from your backpack and heal.  Your health
increased by36hitpoints
The mostner has 2 hitpoints left
You have 92 hitpoints left
You have 96 hit points...the monster attacks you!
```

Figure 10: Added variable to store heal amount so I could use it in text!

The heal button now increases the players health by a number between 30 and 40 however there is currently no restriction on the amount you heal up to.


WindowsFormsApplication11 - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM SQL TOOLS TEST ANALYZE WINDOW HELP
Options.cs | Options.cs [Design] | Menu.cs | Menu.cs [Design] | GameMenu.cs | GameMenu.cs [Design] | Game.cs | Game.cs [Design]
HealButton_Click(object sender, EventArgs e)
Form1
Block
Forwards
Attack
Stats
Left
Use
Right
Inventory
Heal
Back
Menu
Text input
Enter
Solution Explorer

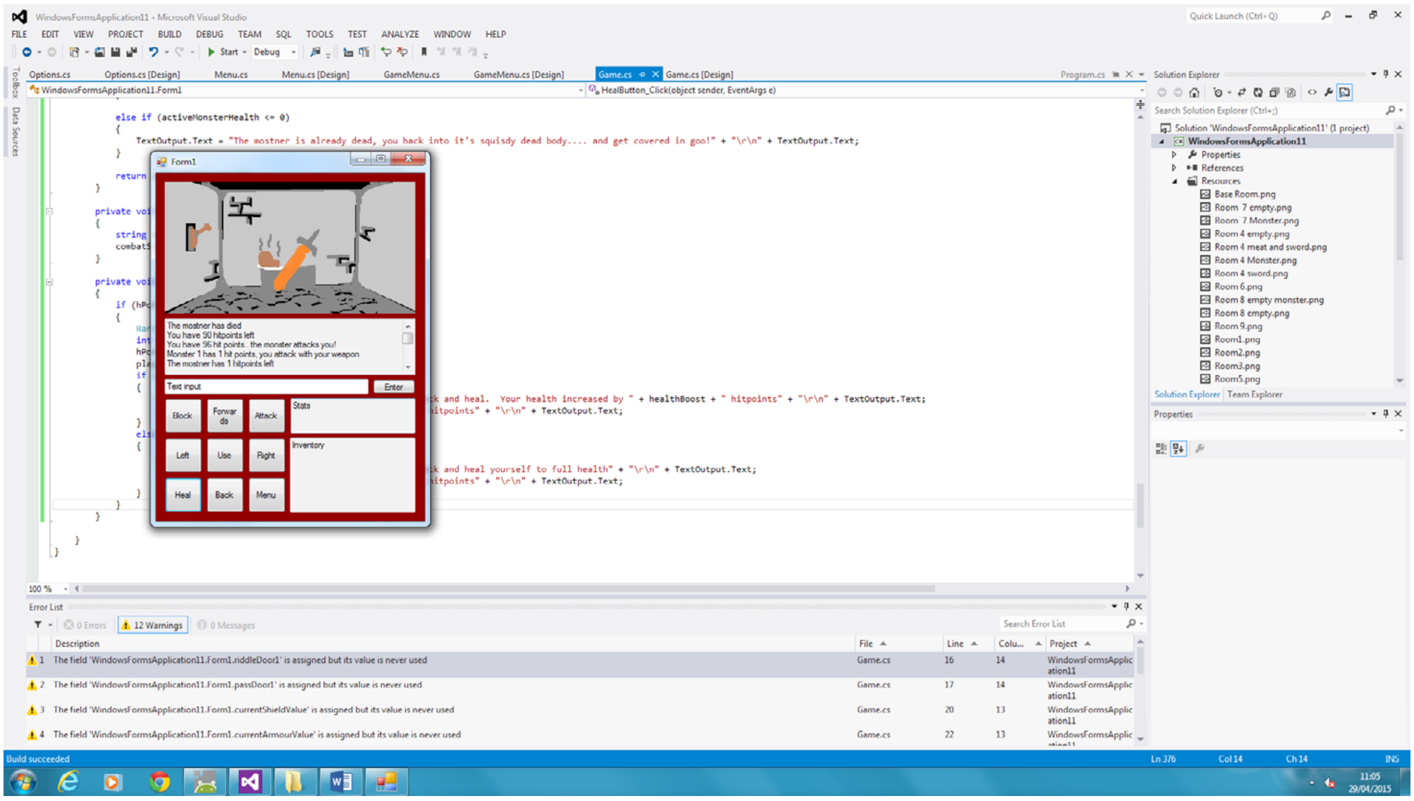
```
ead, you hack into it's squisdy dead body.... and get covered in goo!" + "\r\n" + TextOutput.Text;

Args e)

rgs e)

our backpack and heal.  Your health increased by " + healthBoost + " hitpoints" + "\r\n" + TextOutput.Text;
ealth + "hitpoints" + "\r\n" + TextOutput.Text;
```

```
You now have 133hitpoints
You swig a potion from your backpack and heal.  Your health
increased by 33 hitpoints
Text output
```

```
Error List
0 Errors   12 Warnings   0 Messages
  Description                                                                              File     Line  Colu...  Project
1 The field 'WindowsFormsApplication11.Form1.riddleDoor1' is assigned but its value is never used        Game.cs  16    14       WindowsFormsApplication11
2 The field 'WindowsFormsApplication11.Form1.passDoor1' is assigned but its value is never used          Game.cs  17    14       WindowsFormsApplication11
3 The field 'WindowsFormsApplication11.Form1.currentShieldValue' is assigned but its value is never used Game.cs  20    13       WindowsFormsApplication11
4 The field 'WindowsFormsApplication11.Form1.currentArmourValue' is assigned but its value is never used Game.cs  22    13       WindowsFormsApplication11
Build succeeded                                                             Ln 372   Col 1   Ch 1   INS
```

Figure 11: Health went over 100, added IF statement

I added an IF statement that only allows the players health to reach 100. If the players health is 80 and the player heals for 35 if would only go up to 100 rather than to 115. This was because the health went over 100 sometimes and this is not something that should be happening.

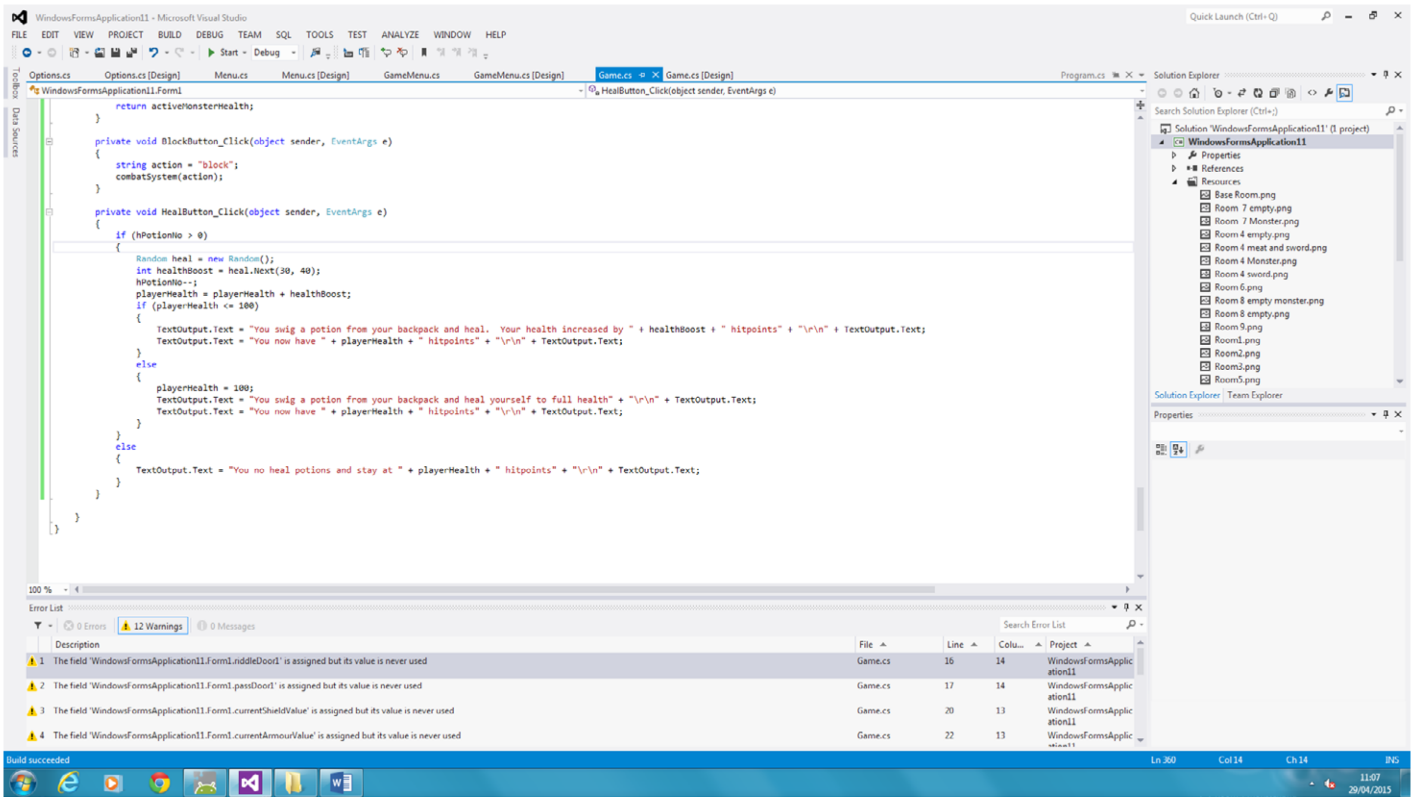

WindowsFormsApplication11 - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM SQL TOOLS TEST ANALYZE WINDOW HELP
Options.cs | Options.cs [Design] | Menu.cs | Menu.cs [Design] | GameMenu.cs | GameMenu.cs [Design] | Game.cs | Game.cs [Design]
WindowsFormsApplication11.Form1
HealButton_Click(object sender, EventArgs e)
Form1
Block
Forwards
Attack
Stats
Left
Use
Right
Inventory
Heal
Back
Menu
Text input
Enter
Solution Explorer

```
            else if (activeMonsterHealth <= 0)
            {
                TextOutput.Text = "The mostner is already dead, you hack into it's squisdy dead body.... and get covered in goo!" + "\r\n" + TextOutput.Text;
            }

            return
        }

        private voi
        {
            string
            combatS
        }

        private voi
        {
            if (hPo
            {
                Ran
                int
                hPo
                pla
                if                     k and heal.  Your health increased by " + healthBoost + " hitpoints" + "\r\n" + TextOutput.Text;
                {                      itpoints" + "\r\n" + TextOutput.Text;

                }
                els
                {                      k and heal yourself to full health" + "\r\n" + TextOutput.Text;
                                       itpoints" + "\r\n" + TextOutput.Text;

                }
            }
        }

    }
}
```

```
The mostner has died
You have 90 hitpoints left
You have 96 hit points...the monster attacks you!
Monster 1 has 1 hit points, you attack with your weapon
The mostner has 1 hitpoints left
```

```
Error List
0 Errors   12 Warnings   0 Messages
  Description                                                                              File     Line  Colu...  Project
1 The field 'WindowsFormsApplication11.Form1.riddleDoor1' is assigned but its value is never used        Game.cs  16    14       WindowsFormsApplication11
2 The field 'WindowsFormsApplication11.Form1.passDoor1' is assigned but its value is never used          Game.cs  17    14       WindowsFormsApplication11
3 The field 'WindowsFormsApplication11.Form1.currentShieldValue' is assigned but its value is never used Game.cs  20    13       WindowsFormsApplication11
4 The field 'WindowsFormsApplication11.Form1.currentArmourValue' is assigned but its value is never used Game.cs  22    13       WindowsFormsApplication11
Build succeeded                                                             Ln 376   Col 14   Ch 14   INS
```

Figure 12: Heal now stops at 100 – need error message to say no heal pots left.

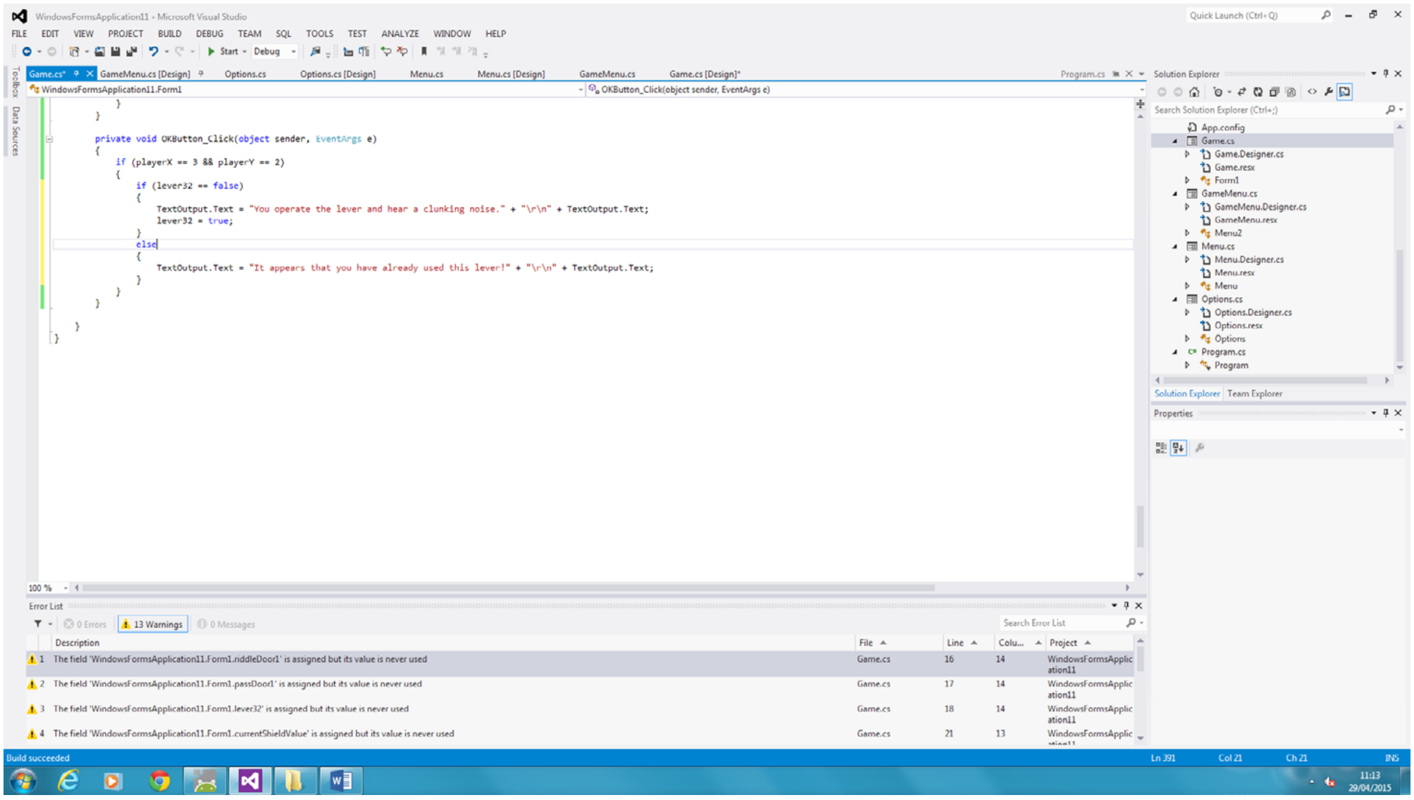


```
            return activeMonsterHealth;
        }

        private void BlockButton_Click(object sender, EventArgs e)
        {
            string action = "block";
            combatSystem(action);
        }

        private void HealButton_Click(object sender, EventArgs e)
        {
            if (hPotionNo > 0)
            {
                Random heal = new Random();
                int healthBoost = heal.Next(30, 40);
                hPotionNo--;
                playerHealth = playerHealth + healthBoost;
                if (playerHealth <= 100)
                {
                    TextOutput.Text = "You swig a potion from your backpack and heal.  Your health increased by " + healthBoost + " hitpoints" + "\r\n" + TextOutput.Text;
                    TextOutput.Text = "You now have " + playerHealth + " hitpoints" + "\r\n" + TextOutput.Text;
                }
                else
                {
                    playerHealth = 100;
                    TextOutput.Text = "You swig a potion from your backpack and heal yourself to full health" + "\r\n" + TextOutput.Text;
                    TextOutput.Text = "You now have " + playerHealth + " hitpoints" + "\r\n" + TextOutput.Text;
                }
            }
            else
            {
                TextOutput.Text = "You no heal potions and stay at " + playerHealth + " hitpoints" + "\r\n" + TextOutput.Text;
            }
        }

    }
}
```

	Description	File	Line	Colu...	Project
1	The field 'WindowsFormsApplication11.Form1.riddleDoor1' is assigned but its value is never used	Game.cs	16	14	WindowsFormsApplication11
2	The field 'WindowsFormsApplication11.Form1.passDoor1' is assigned but its value is never used	Game.cs	17	14	WindowsFormsApplication11
3	The field 'WindowsFormsApplication11.Form1.currentShieldValue' is assigned but its value is never used	Game.cs	20	13	WindowsFormsApplication11
4	The field 'WindowsFormsApplication11.Form1.currentArmourValue' is assigned but its value is never used	Game.cs	22	13	WindowsFormsApplic

Figure 13: solution

I created a method for the use button so that when it is clicked, the players current coordinates are compared to IF statements to find out what the player is "using".

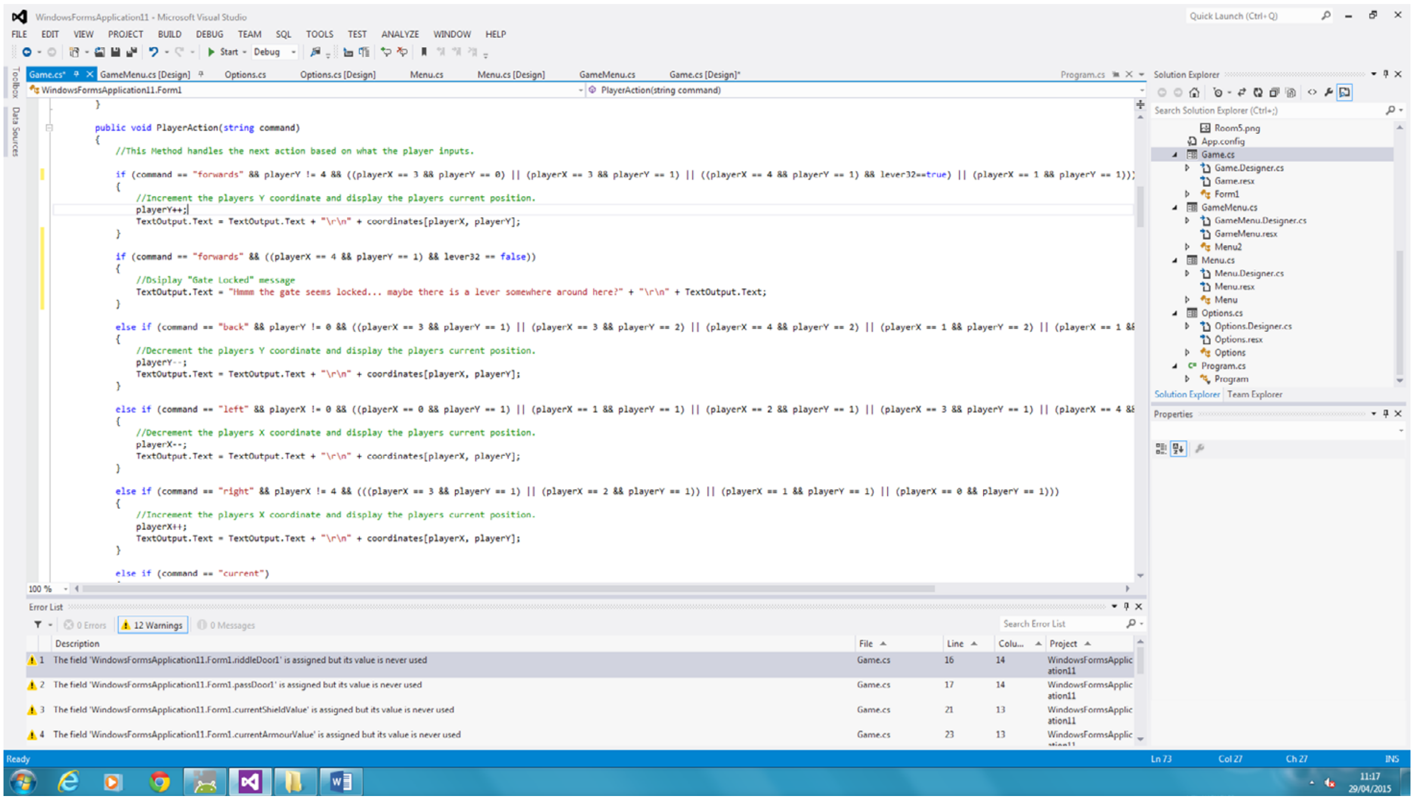


```
            }
        }

        private void OKButton_Click(object sender, EventArgs e)
        {
            if (playerX == 3 && playerY == 2)
            {
                if (lever32 == false)
                {
                    TextOutput.Text = "You operate the lever and hear a clunking noise." + "\r\n" + TextOutput.Text;
                    lever32 = true;
                }
                else
                {
                    TextOutput.Text = "It appears that you have already used this lever!" + "\r\n" + TextOutput.Text;
                }
            }
        }

    }
}
```

	Description	File	Line	Colu...	Project
1	The field 'WindowsFormsApplication11.Form1.riddleDoor1' is assigned but its value is never used	Game.cs	16	14	WindowsFormsApplication11
2	The field 'WindowsFormsApplication11.Form1.passDoor1' is assigned but its value is never used	Game.cs	17	14	WindowsFormsApplication11
3	The field 'WindowsFormsApplication11.Form1.lever32' is assigned but its value is never used	Game.cs	18	14	WindowsFormsApplication11
4	The field 'WindowsFormsApplication11.Form1.currentShieldValue' is assigned but its value is never used	Game.cs	21	13	WindowsFormsApplic

Figure 14: Coding USE button. Set of IF statements to control which levers have been used in which rooms

In order to stop the player moving through the gate into room (4, 2) I had to add another logic statement to the IF statement for the command "forwards" so that the player can only move forwards in room (4, 1) if lever32 is true (If the lever has been activated).



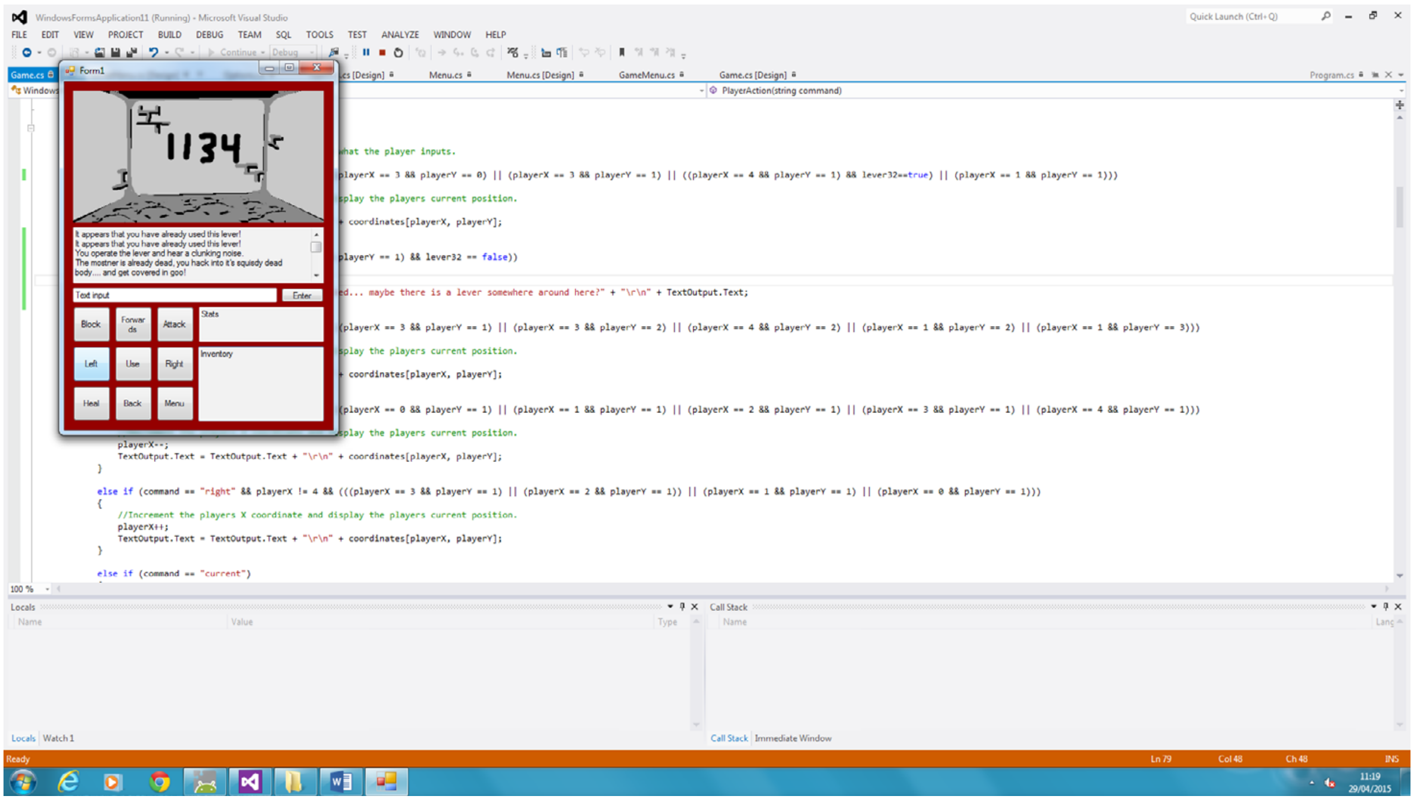
```
        }

        public void PlayerAction(string command)
        {
            //This Method handles the next action based on what the player inputs.

            if (command == "forwards" && playerY != 4 && ((playerX == 3 && playerY == 0) || (playerX == 3 && playerY == 1) || ((playerX == 4 && playerY == 1) && lever32==true) || (playerX == 1 && playerY == 1))
            {
                //Increment the players Y coordinate and display the players current position.
                playerY++;
                TextOutput.Text = TextOutput.Text + "\r\n" + coordinates[playerX, playerY];
            }

            if (command == "forwards" && ((playerX == 4 && playerY == 1) && lever32 == false))
            {
                //Dsiplay "Gate Locked" message
                TextOutput.Text = "Hmmm the gate seems locked... maybe there is a lever somewhere around here?" + "\r\n" + TextOutput.Text;
            }

            else if (command == "back" && playerY != 0 && ((playerX == 3 && playerY == 1) || (playerX == 3 && playerY == 2) || (playerX == 4 && playerY == 2) || (playerX == 1 && playerY == 2) || (playerX == 1 &&
            {
                //Decrement the players Y coordinate and display the players current position.
                playerY--;
                TextOutput.Text = TextOutput.Text + "\r\n" + coordinates[playerX, playerY];
            }

            else if (command == "left" && playerX != 0 && ((playerX == 0 && playerY == 1) || (playerX == 1 && playerY == 1) || (playerX == 2 && playerY == 1) || (playerX == 3 && playerY == 1) || (playerX == 4 &&
            {
                //Decrement the players X coordinate and display the players current position.
                playerX--;
                TextOutput.Text = TextOutput.Text + "\r\n" + coordinates[playerX, playerY];
            }

            else if (command == "right" && playerX != 4 && (((playerX == 3 && playerY == 1) || (playerX == 2 && playerY == 1)) || (playerX == 1 && playerY == 1) || (playerX == 0 && playerY == 1)))
            {
                //Increment the players X coordinate and display the players current position.
                playerX++;
                TextOutput.Text = TextOutput.Text + "\r\n" + coordinates[playerX, playerY];
            }

            else if (command == "current")
```

Error List — 0 Errors | 12 Warnings | 0 Messages

	Description	File	Line	Colu...	Project
1	The field 'WindowsFormsApplication11.Form1.riddleDoor1' is assigned but its value is never used	Game.cs	16	14	WindowsFormsApplication11
2	The field 'WindowsFormsApplication11.Form1.passDoor1' is assigned but its value is never used	Game.cs	17	14	WindowsFormsApplication11
3	The field 'WindowsFormsApplication11.Form1.currentShieldValue' is assigned but its value is never used	Game.cs	21	13	WindowsFormsApplication11
4	The field 'WindowsFormsApplication11.Form1.currentArmourValue' is assigned but its value is never used	Game.cs	23	13	WindowsFormsApplication11

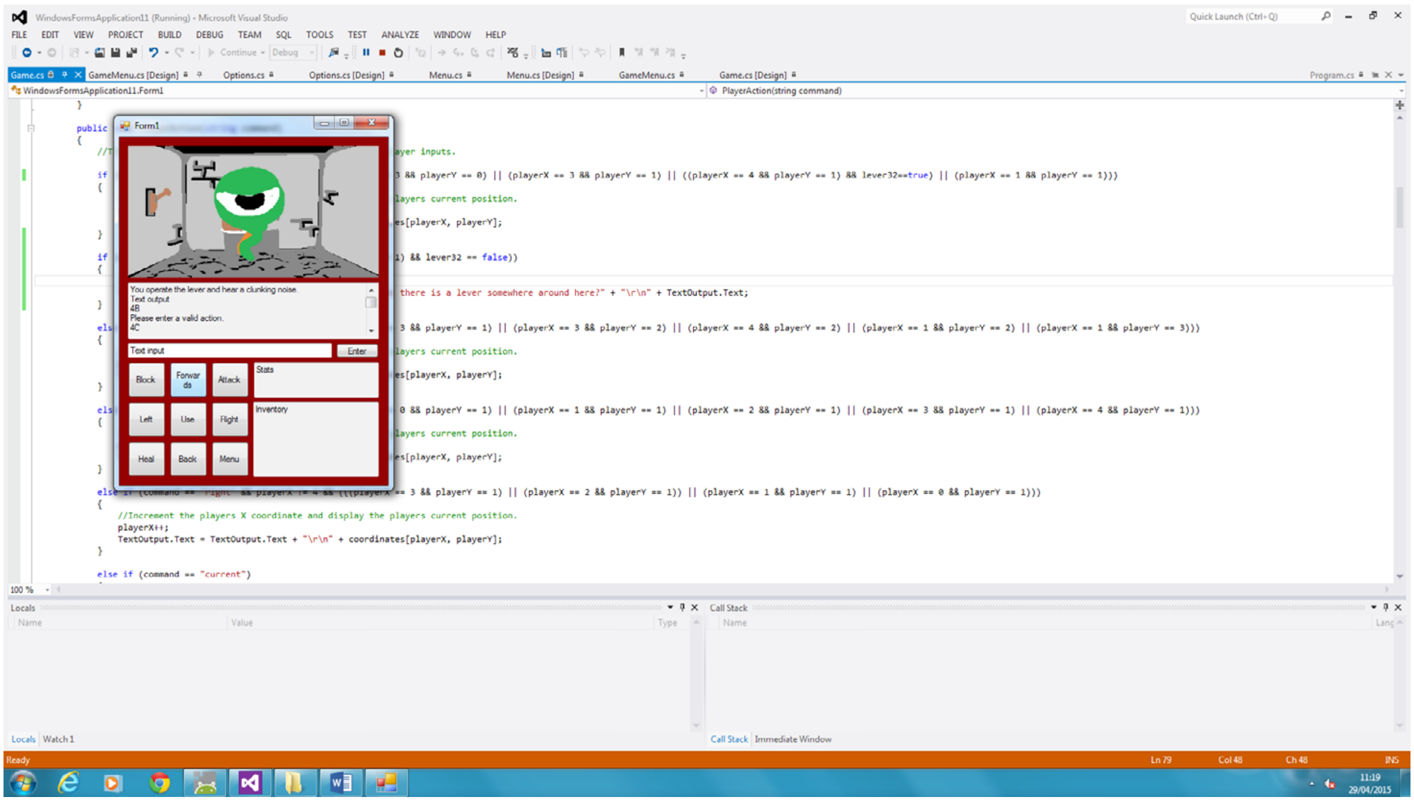
Figure 15: Added logic statements to movement to stop people moving through the locked door linked to Room (3, 2) Lever

I ran a quick test to see if the gate in room (4, 1) was working as it should. After testing it with and without the lever being active, show test showed that the code was working as intended with just one problem. The lever could be activated without having defeated the monster in the room but I wanted the lever to only be usable after defeating the monster.



```
It appears that you have already used this lever!
It appears that you have already used this lever!
You operate the lever and hear a clunking noise.
The mostner is already dead, you hack into it's squisdy dead
body.... and get covered in goo!
```

Figure 16: lever check system test


```
        }

        public
        {
            //T                                                  ayer inputs.

            if                                                   3 && playerY == 0) || (playerX == 3 && playerY == 1) || ((playerX == 4 && playerY == 1) && lever32==true) || (playerX == 1 && playerY == 1)))
            {
                                                                 layers current position.

                                                                 es[playerX, playerY];
            }

            if                                                   1) && lever32 == false))
            {
                                                                 there is a lever somewhere around here?" + "\r\n" + TextOutput.Text;
            }

            els                                                  3 && playerY == 1) || (playerX == 3 && playerY == 2) || (playerX == 4 && playerY == 2) || (playerX == 1 && playerY == 2) || (playerX == 1 && playerY == 3)))
            {
                                                                 layers current position.

                                                                 es[playerX, playerY];
            }

            els                                                  0 && playerY == 1) || (playerX == 1 && playerY == 1) || (playerX == 2 && playerY == 1) || (playerX == 3 && playerY == 1) || (playerX == 4 && playerY == 1)))
            {
                                                                 layers current position.

                                                                 es[playerX, playerY];
            }

            else                                                 == 3 && playerY == 1) || (playerX == 2 && playerY == 1)) || (playerX == 1 && playerY == 1) || (playerX == 0 && playerY == 1)))
            {
                //Increment the players X coordinate and display the players current position.
                playerX++;
                TextOutput.Text = TextOutput.Text + "\r\n" + coordinates[playerX, playerY];
            }

            else if (command == "current")
```

```
You operate the lever and hear a clunking noise.
Text output
4B
Please enter a valid action.
4C
```

Figure 17: Bug – you can use level before monster is dead

To fix the lever problem I added some if statements. If the player uses the lever and the enemy is alive then the player is informed that they must first defeat the enemy, however if the enemy is dead then the player is informed that they have pulled the lever and the lever is activated.

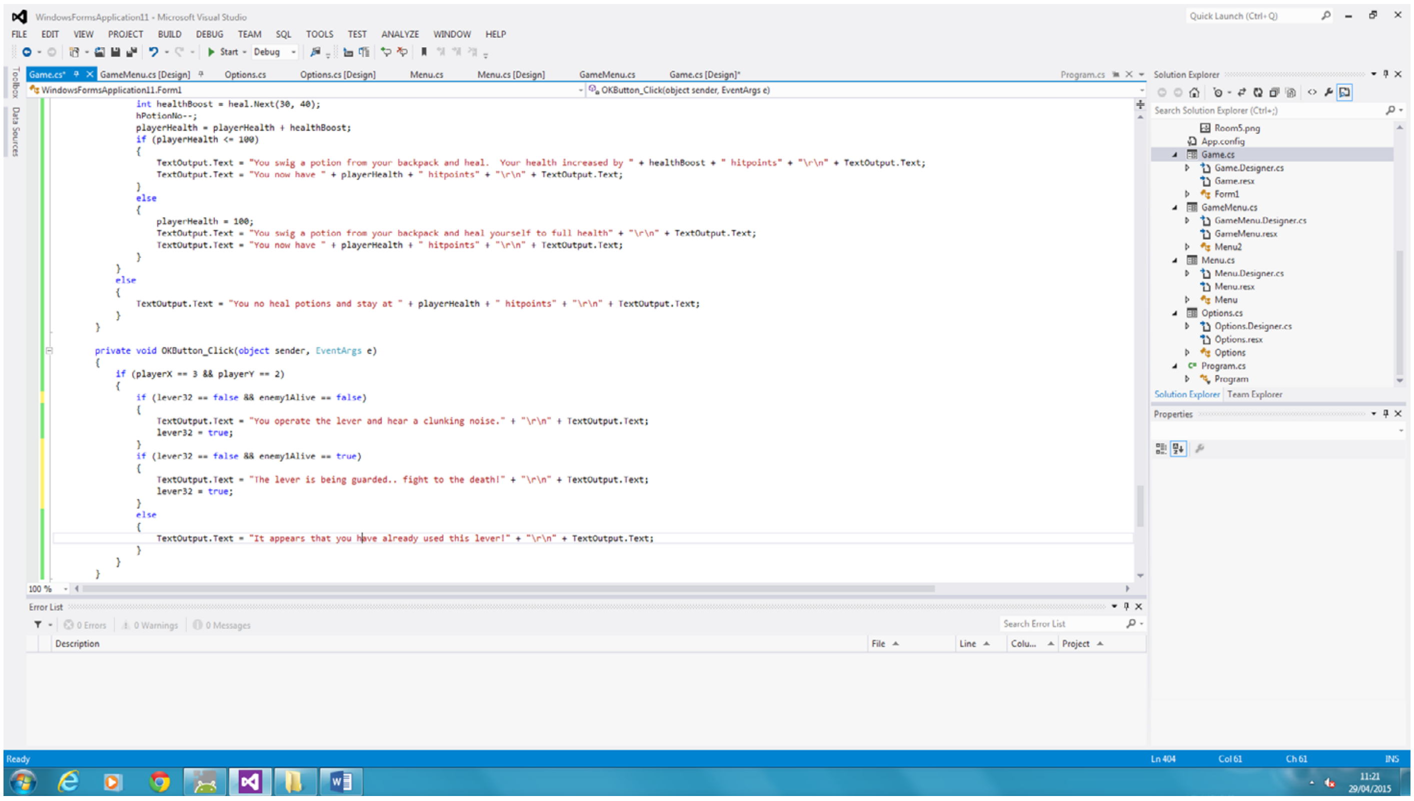
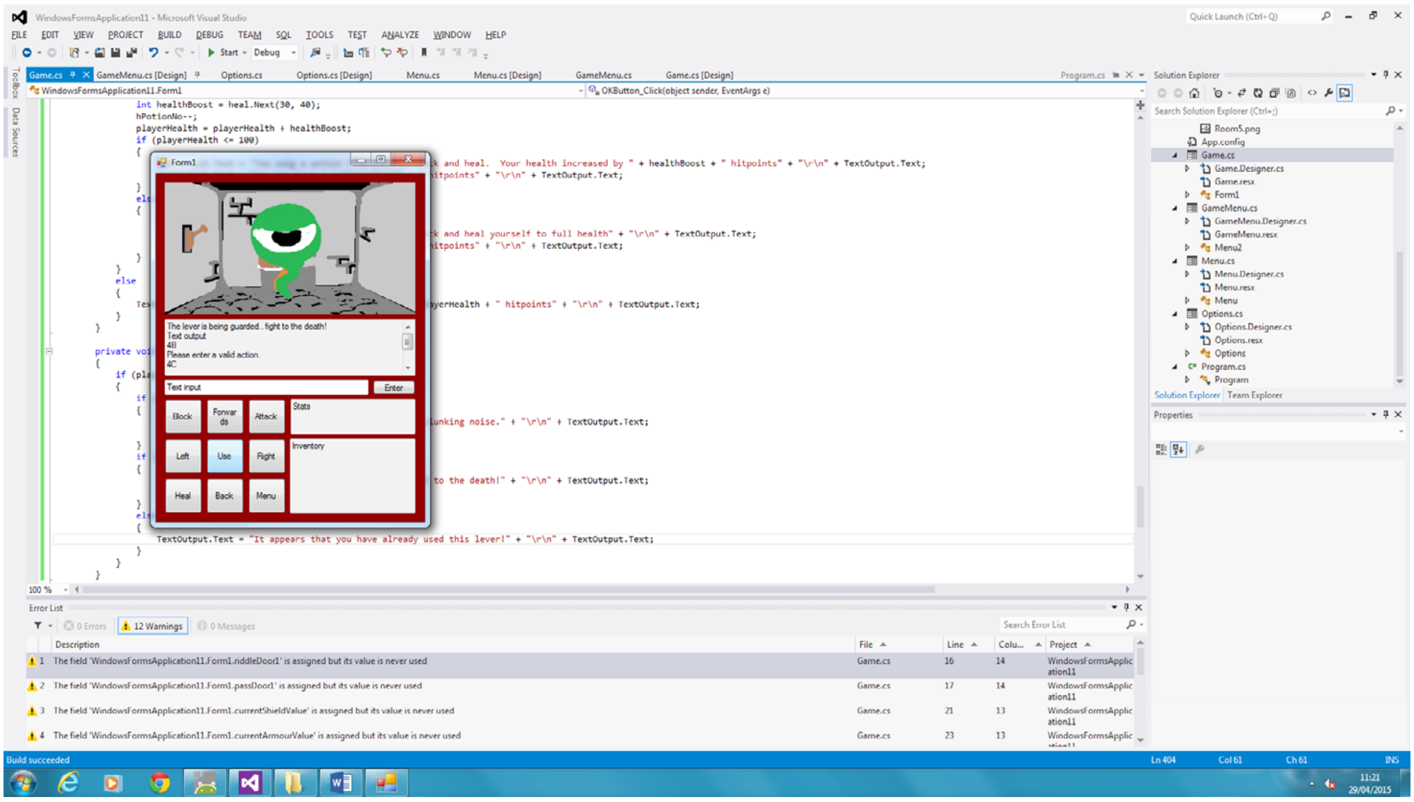

```
                int healthBoost = heal.Next(30, 40);
                hPotionNo--;
                playerHealth = playerHealth + healthBoost;
                if (playerHealth <= 100)
                {
                    TextOutput.Text = "You swig a potion from your backpack and heal.  Your health increased by " + healthBoost + " hitpoints" + "\r\n" + TextOutput.Text;
                    TextOutput.Text = "You now have " + playerHealth + " hitpoints" + "\r\n" + TextOutput.Text;
                }
                else
                {
                    playerHealth = 100;
                    TextOutput.Text = "You swig a potion from your backpack and heal yourself to full health" + "\r\n" + TextOutput.Text;
                    TextOutput.Text = "You now have " + playerHealth + " hitpoints" + "\r\n" + TextOutput.Text;
                }
            }
            else
            {
                TextOutput.Text = "You no heal potions and stay at " + playerHealth + " hitpoints" + "\r\n" + TextOutput.Text;
            }
        }

        private void OKButton_Click(object sender, EventArgs e)
        {
            if (playerX == 3 && playerY == 2)
            {
                if (lever32 == false && enemy1Alive == false)
                {
                    TextOutput.Text = "You operate the lever and hear a clunking noise." + "\r\n" + TextOutput.Text;
                    lever32 = true;
                }
                if (lever32 == false && enemy1Alive == true)
                {
                    TextOutput.Text = "The lever is being guarded.. fight to the death!" + "\r\n" + TextOutput.Text;
                    lever32 = true;
                }
                else
                {
                    TextOutput.Text = "It appears that you have already used this lever!" + "\r\n" + TextOutput.Text;
                }
            }
        }
```

Figure 18: Added logic to prevent lever being used whilst monster is alive.



```
                int healthBoost = heal.Next(30, 40);
                hPotionNo--;
                playerHealth = playerHealth + healthBoost;
                if (playerHealth <= 100)
                {
                                    k and heal.  Your health increased by " + healthBoost + " hitpoints" + "\r\n" + TextOutput.Text;
                                    itpoints" + "\r\n" + TextOutput.Text;
                }
                els
                {
                                    k and heal yourself to full health" + "\r\n" + TextOutput.Text;
                                    itpoints" + "\r\n" + TextOutput.Text;
                }
            }
            else
            {
                Tex                 yerHealth + " hitpoints" + "\r\n" + TextOutput.Text;
            }
        }

        private voi
        {
            if (pla
            {
                if
                {
                                    lunking noise." + "\r\n" + TextOutput.Text;
                }
                if
                {
                                    to the death!" + "\r\n" + TextOutput.Text;
                }
                els
                {
                    TextOutput.Text = "It appears that you have already used this lever!" + "\r\n" + TextOutput.Text;
                }
            }
        }
```

```
The lever is being guarded.. fight to the death!
Text output
4B
Please enter a valid action.
4C
```

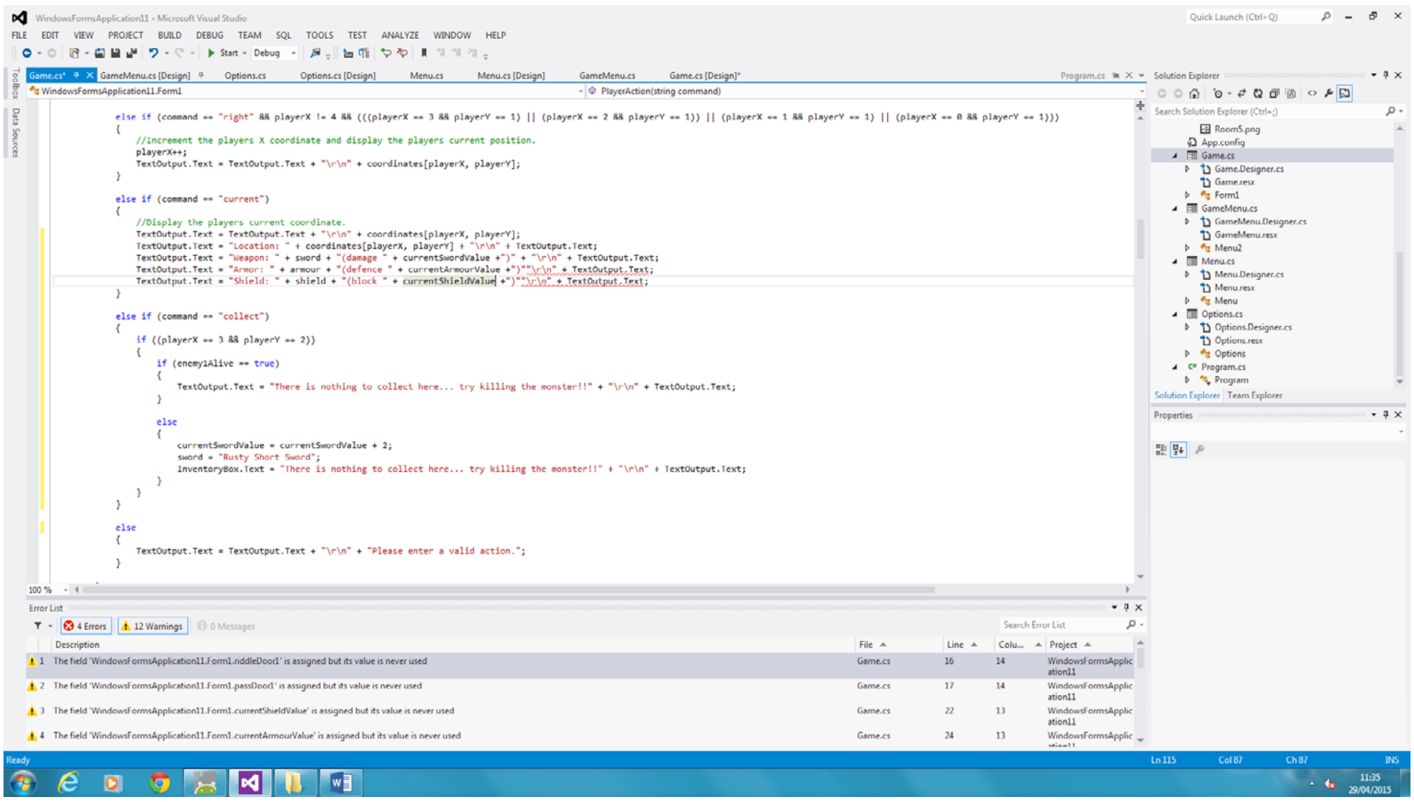
```
Error List
0 Errors | 12 Warnings | 0 Messages

   Description                                                                                 File     Line  Colu...  Project
1  The field 'WindowsFormsApplication11.Form1.riddleDoor1' is assigned but its value is never used          Game.cs  16    14       WindowsFormsApplication11
2  The field 'WindowsFormsApplication11.Form1.passDoor1' is assigned but its value is never used            Game.cs  17    14       WindowsFormsApplication11
3  The field 'WindowsFormsApplication11.Form1.currentShieldValue' is assigned but its value is never used   Game.cs  21    13       WindowsFormsApplication11
4  The field 'WindowsFormsApplication11.Form1.currentArmourValue' is assigned but its value is never used   Game.cs  23    13       WindowsFormsApplic

Build succeeded
```

Figure 19: Lever Check evidence

I then decided it would be best to move to working on how the player will collect items.

I will create an array for the player's inventory so that I can set a limit on how many items can be carried at a time. Even if I don't end up setting a limit, having an array to store inventory items makes things a little easier.

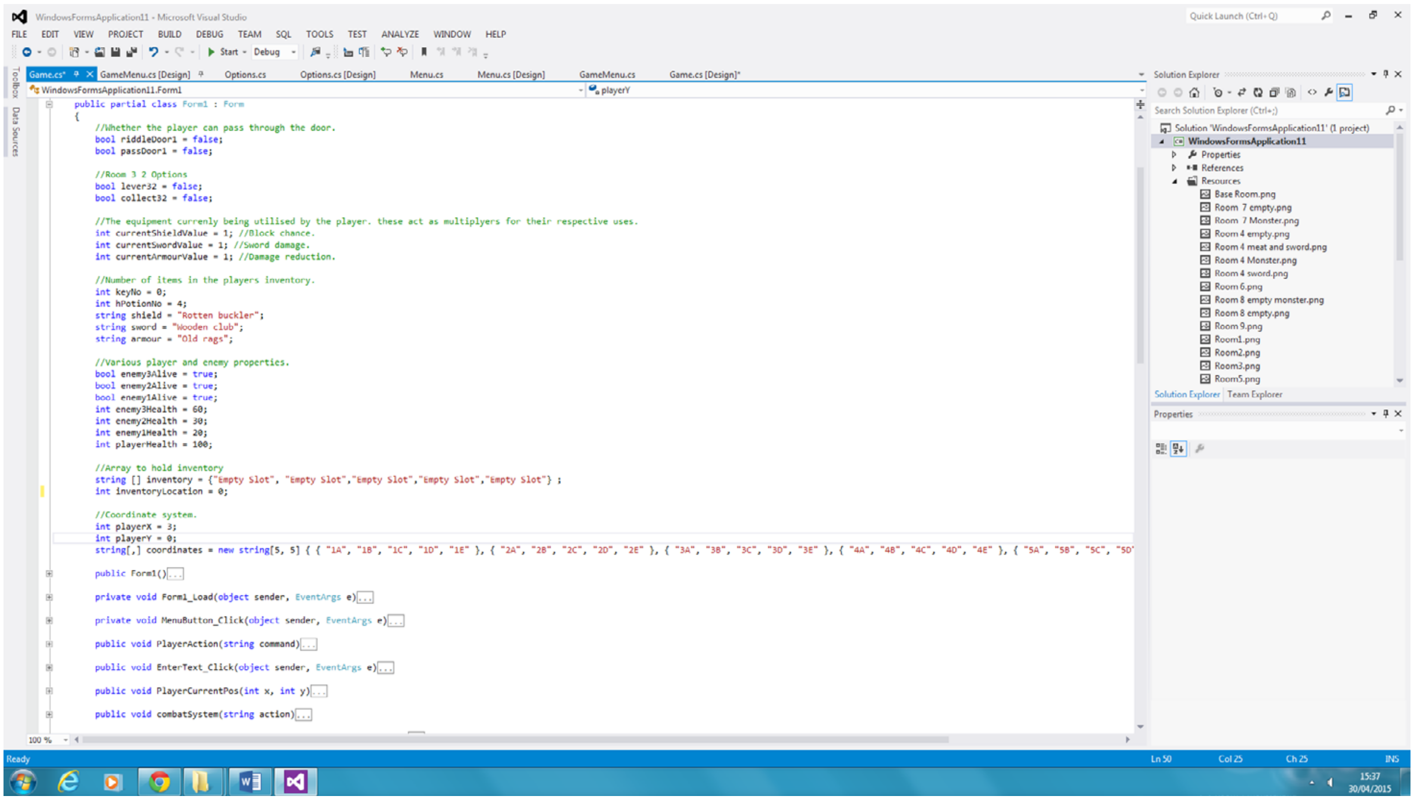
I Changed the IF statement within the "playerAction" method that runs when the command "current" is input so that as well as displaying the current location of the player, the current sword, shield and armour are displayed along with their values. I also created a new IF statement for the command "collect" which will retrieve any items in the current room as long as there are no enemies alive in the room.



```
else if (command == "right" && playerX != 4 && (((playerX == 3 && playerY == 1) || (playerX == 2 && playerY == 1)) || (playerX == 1 && playerY == 1) || (playerX == 0 && playerY == 1)))
{
    //Increment the players X coordinate and display the players current position.
    playerX++;
    TextOutput.Text = TextOutput.Text + "\r\n" + coordinates[playerX, playerY];
}

else if (command == "current")
{
    //Display the players current coordinate.
    TextOutput.Text = TextOutput.Text + "\r\n" + coordinates[playerX, playerY];
    TextOutput.Text = "Location: " + coordinates[playerX, playerY] + "\r\n" + TextOutput.Text;
    TextOutput.Text = "Weapon: " + sword + "(damage " + currentSwordValue +")" + "\r\n" + TextOutput.Text;
    TextOutput.Text = "Armor: " + armour + "(defence " + currentArmourValue +")""\r\n" + TextOutput.Text;
    TextOutput.Text = "Shield: " + shield + "(block " + currentShieldValue +")""\r\n" + TextOutput.Text;
}

else if (command == "collect")
{
    if ((playerX == 3 && playerY == 2))
    {
        if (enemy1Alive == true)
        {
            TextOutput.Text = "There is nothing to collect here... try killing the monster!!" + "\r\n" + TextOutput.Text;
        }

        else
        {
            currentSwordValue = currentSwordValue + 2;
            sword = "Rusty Short Sword";
            InventoryBox.Text = "There is nothing to collect here... try killing the monster!!" + "\r\n" + TextOutput.Text;
        }
    }
}

else
{
    TextOutput.Text = TextOutput.Text + "\r\n" + "Please enter a valid action.";
}
```

```
Error List
4 Errors   12 Warnings   0 Messages

  Description                                                                              File     Line  Colu...  Project
1 The field 'WindowsFormsApplication11.Form1.riddleDoor1' is assigned but its value is never used     Game.cs  16    14       WindowsFormsApplication11
2 The field 'WindowsFormsApplication11.Form1.passDoor1' is assigned but its value is never used       Game.cs  17    14       WindowsFormsApplication11
3 The field 'WindowsFormsApplication11.Form1.currentShieldValue' is assigned but its value is never used  Game.cs  22  13    WindowsFormsApplication11
4 The field 'WindowsFormsApplication11.Form1.currentArmourValue' is assigned but its value is never used  Game.cs  24  13    WindowsFormsApplic
```

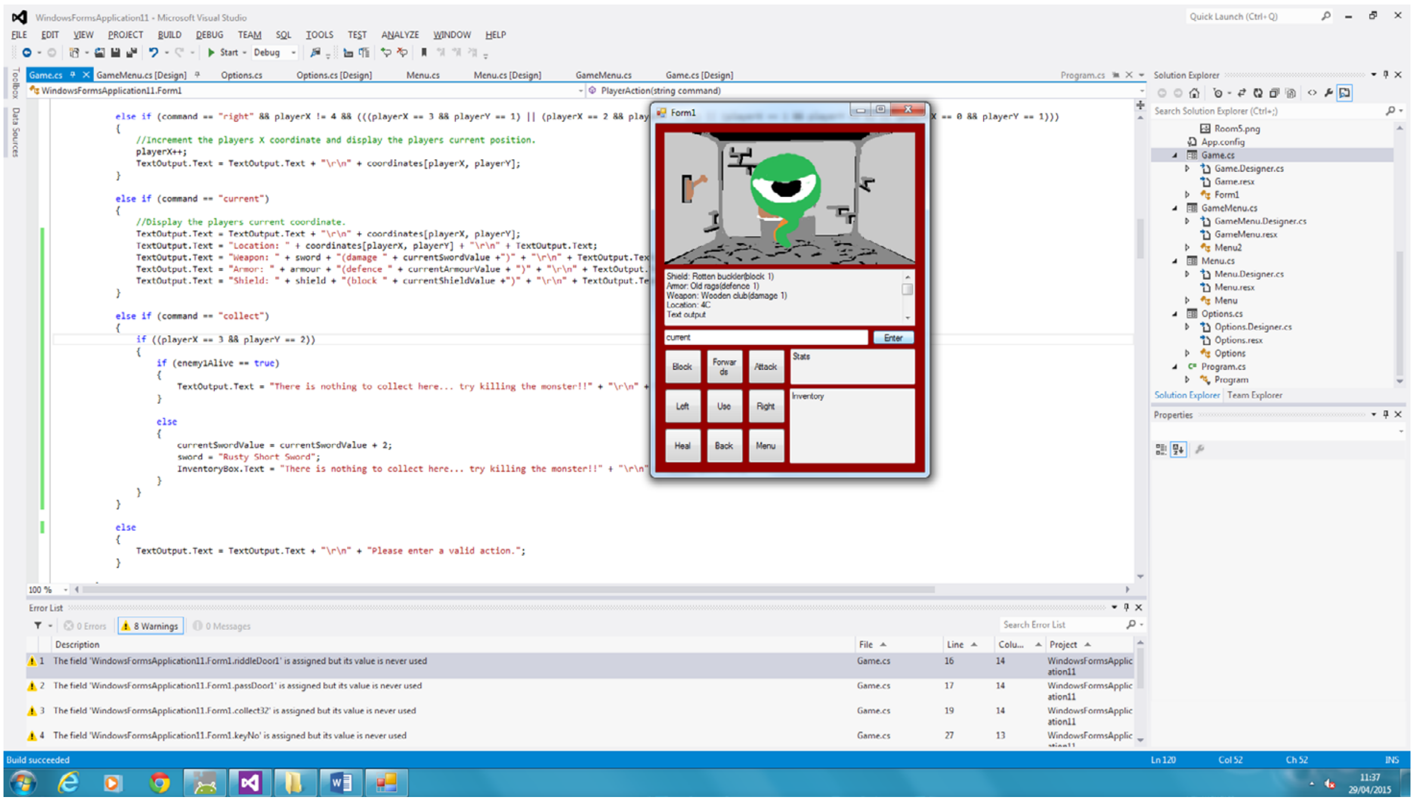
Figure 20: Added code to collect items from room with IF statement based on monster being dead and updated "Current" to include values of all items.



```
public partial class Form1 : Form
{
    //Whether the player can pass through the door.
    bool riddleDoor1 = false;
    bool passDoor1 = false;

    //Room 3 2 Options
    bool lever32 = false;
    bool collect32 = false;

    //The equipment currenly being utilised by the player. these act as multiplyers for their respective uses.
    int currentShieldValue = 1; //Block chance.
    int currentSwordValue = 1; //Sword damage.
    int currentArmourValue = 1; //Damage reduction.

    //Number of items in the players inventory.
    int keyNo = 0;
    int hPotionNo = 4;
    string shield = "Rotten buckler";
    string sword = "Wooden club";
    string armour = "Old rags";

    //Various player and enemy properties.
    bool enemy3Alive = true;
    bool enemy2Alive = true;
    bool enemy1Alive = true;
    int enemy3Health = 60;
    int enemy2Health = 30;
    int enemy1Health = 20;
    int playerHealth = 100;

    //Array to hold inventory
    string [] inventory = {"Empty Slot", "Empty Slot","Empty Slot","Empty Slot","Empty Slot"} ;
    int inventoryLocation = 0;

    //Coordinate system.
    int playerX = 3;
    int playerY = 0;
    string[,] coordinates = new string[5, 5] { { "1A", "1B", "1C", "1D", "1E" }, { "2A", "2B", "2C", "2D", "2E" }, { "3A", "3B", "3C", "3D", "3E" }, { "4A", "4B", "4C", "4D", "4E" }, { "5A", "5B", "5C", "5D"

    public Form1()...

    private void Form1_Load(object sender, EventArgs e)...

    private void MenuButton_Click(object sender, EventArgs e)...

    public void PlayerAction(string command)...

    public void EnterText_Click(object sender, EventArgs e)...

    public void PlayerCurrentPos(int x, int y)...

    public void combatSystem(string action)...
```

Figure 21: I created an array to store the players inventory items as strings.

I ran a test for the "current" and "collect" commands. The "current" command code worked as expected and displayed each element correctly however the "collect" command took the text from the output box and added it to the inventory box. The player is also informed that the monster is still alive and cannot pick up the items when the enemy is defeated.

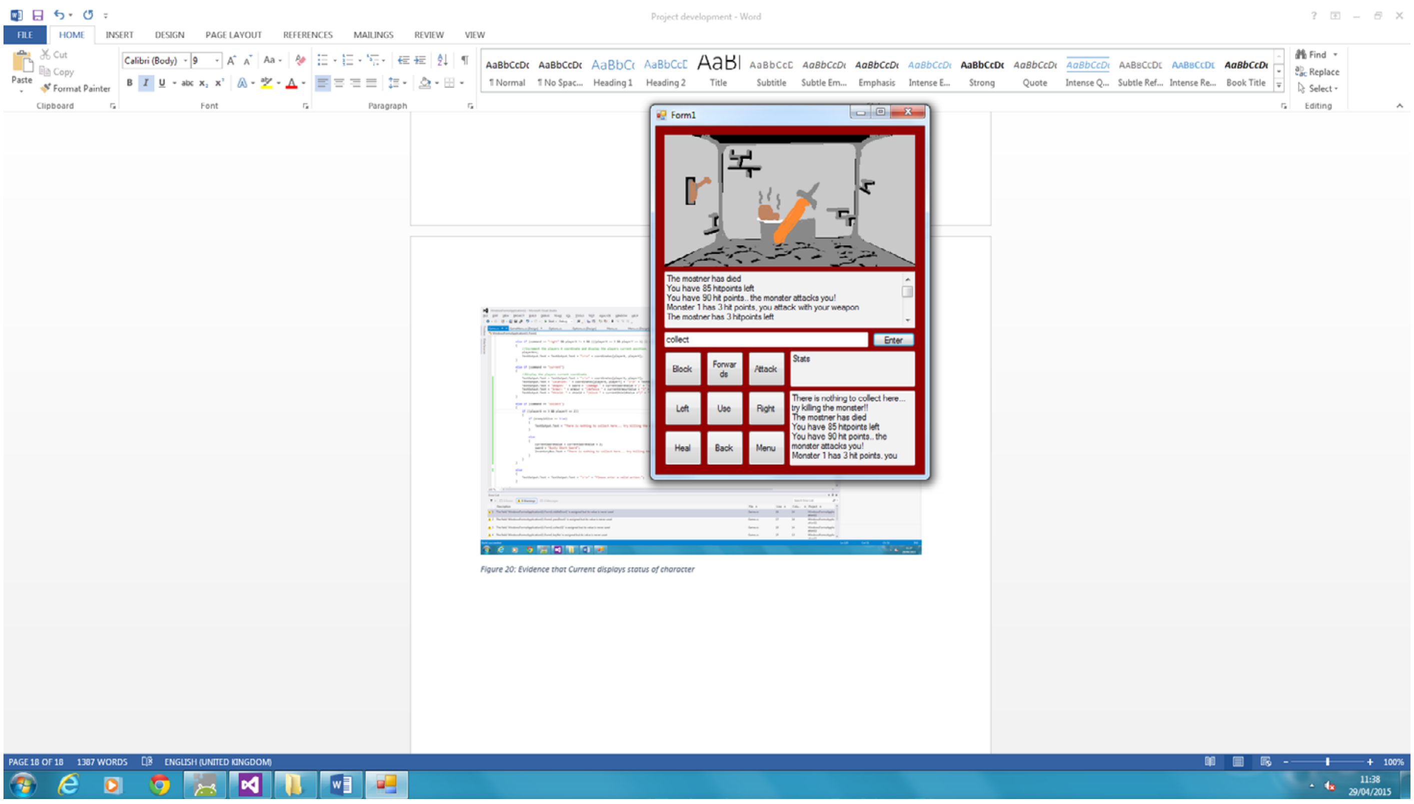


```
else if (command == "right" && playerX != 4 && (((playerX == 3 && playerY == 1) || (playerX == 2 && play          X == 0 && playerY == 1)))
{
    //Increment the players X coordinate and display the players current position.
    playerX++;
    TextOutput.Text = TextOutput.Text + "\r\n" + coordinates[playerX, playerY];
}

else if (command == "current")
{
    //Display the players current coordinate.
    TextOutput.Text = TextOutput.Text + "\r\n" + coordinates[playerX, playerY];
    TextOutput.Text = "Location: " + coordinates[playerX, playerY] + "\r\n" + TextOutput.Text;
    TextOutput.Text = "Weapon: " + sword + "(damage " + currentSwordValue +")" + "\r\n" + TextOutput.Tex
    TextOutput.Text = "Armor: " + armour + "(defence " + currentArmourValue + ")" + "\r\n" + TextOutput.
    TextOutput.Text = "Shield: " + shield + "(block " + currentShieldValue +")" + "\r\n" + TextOutput.Te
}

else if (command == "collect")
{
    if ((playerX == 3 && playerY == 2))
    {
        if (enemy1Alive == true)
        {
            TextOutput.Text = "There is nothing to collect here... try killing the monster!!" + "\r\n" +
        }

        else
        {
            currentSwordValue = currentSwordValue + 2;
            sword = "Rusty Short Sword";
            InventoryBox.Text = "There is nothing to collect here... try killing the monster!!" + "\r\n"
        }
    }
}

else
{
    TextOutput.Text = TextOutput.Text + "\r\n" + "Please enter a valid action.";
}
```

```
Shield: Rotten buckler(block 1)
Armor: Old rags(defence 1)
Weapon: Wooden club(damage 1)
Location: 4C
Text output
```

	Description	File	Line	Colu...	Project
1	The field 'WindowsFormsApplication11.Form1.riddleDoor1' is assigned but its value is never used	Game.cs	16	14	WindowsFormsApplication11
2	The field 'WindowsFormsApplication11.Form1.passDoor1' is assigned but its value is never used	Game.cs	17	14	WindowsFormsApplication11
3	The field 'WindowsFormsApplication11.Form1.collect32' is assigned but its value is never used	Game.cs	19	14	WindowsFormsApplication11
4	The field 'WindowsFormsApplication11.Form1.keyNo' is assigned but its value is never used	Game.cs	27	13	WindowsFormsApplication11

Figure 22: Evidence that Current displays status of character



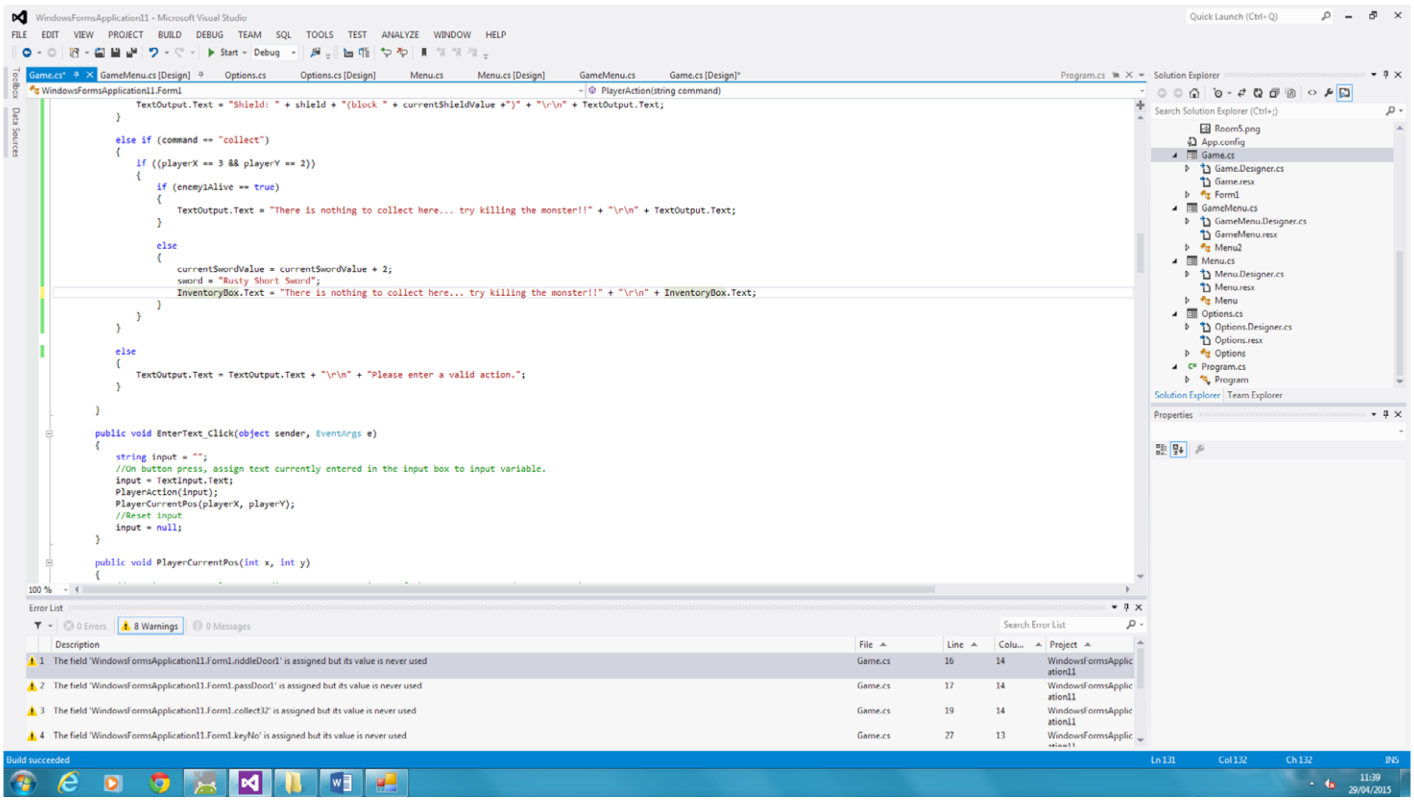
```
The mostner has died
You have 85 hitpoints left
You have 90 hit points.. the monster attacks you!
Monster 1 has 3 hit points, you attack with your weapon
The mostner has 3 hitpoints left
```

```
There is nothing to collect here...
try killing the monster!!
The mostner has died
You have 85 hitpoints left
You have 90 hit points.. the
monster attacks you!
Monster 1 has 3 hit points, you
```

Project development - Word
FILE HOME INSERT DESIGN PAGE LAYOUT REFERENCES MAILINGS REVIEW VIEW
Form1
collect
Enter
Block
Forwards
Attack
Stats
Left
Use
Right
Heal
Back
Menu
Figure 20: Evidence that Current displays status of character
PAGE 18 OF 18
1387 WORDS
ENGLISH (UNITED KINGDOM)

Figure 23: Tried collect after monster dead – fail!

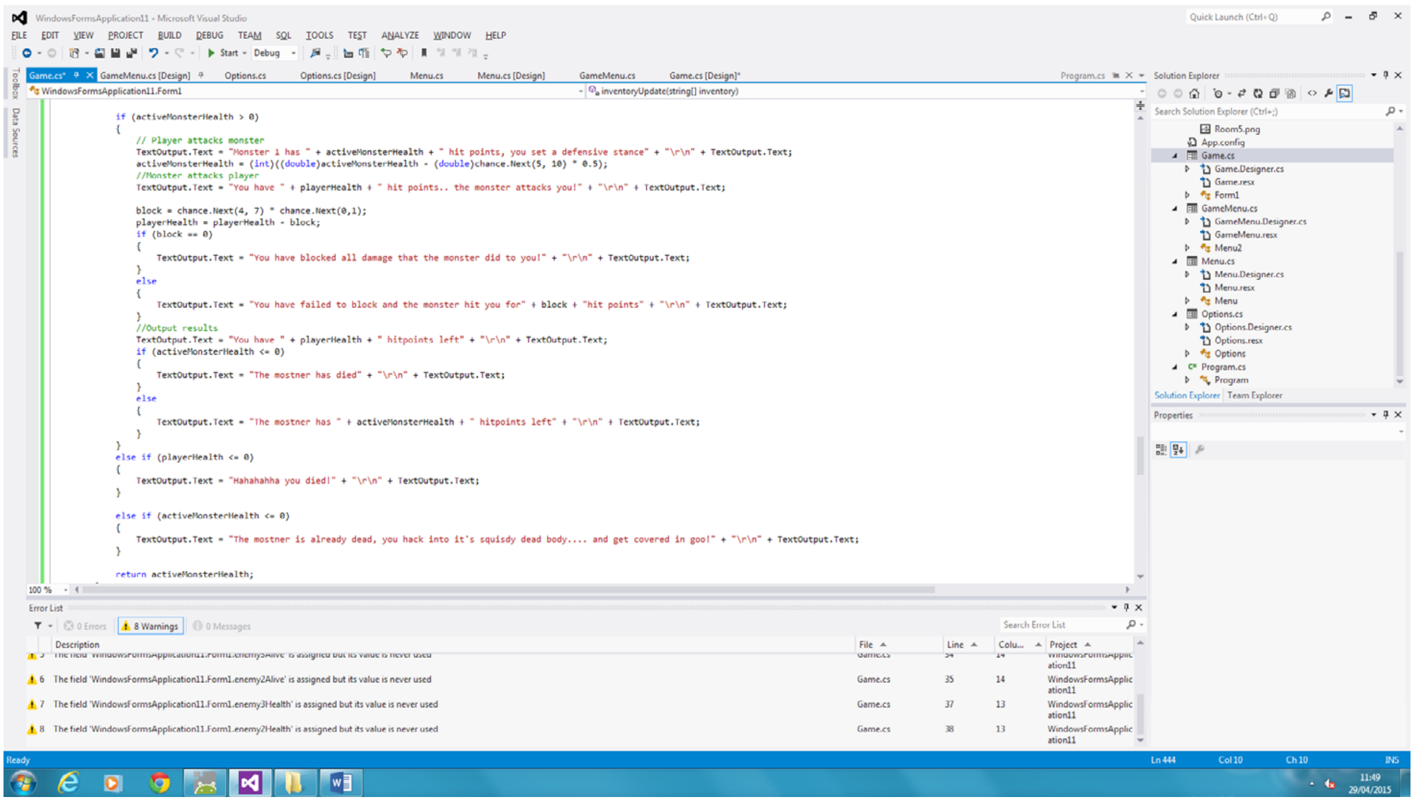
The problem with the text was that the code was setting the inventory box text equal to the message to the player plus the output box's text. To fix this I simply changed output box to inventory box.



```
                TextOutput.Text = "Shield: " + shield + "(block " + currentShieldValue +")" + "\r\n" + TextOutput.Text;
            }

            else if (command == "collect")
            {
                if ((playerX == 3 && playerY == 2))
                {
                    if (enemy1Alive == true)
                    {
                        TextOutput.Text = "There is nothing to collect here... try killing the monster!!" + "\r\n" + TextOutput.Text;
                    }

                    else
                    {
                        currentSwordValue = currentSwordValue + 2;
                        sword = "Rusty Short Sword";
                        InventoryBox.Text = "There is nothing to collect here... try killing the monster!!" + "\r\n" + InventoryBox.Text;
                    }
                }
            }

            else
            {
                TextOutput.Text = TextOutput.Text + "\r\n" + "Please enter a valid action.";
            }

        }

        public void EnterText_Click(object sender, EventArgs e)
        {
            string input = "";
            //On button press, assign text currently entered in the input box to input variable.
            input = TextInput.Text;
            PlayerAction(input);
            PlayerCurrentPos(playerX, playerY);
            //Reset input
            input = null;
        }

        public void PlayerCurrentPos(int x, int y)
        {
```

```
Error List
0 Errors   8 Warnings   0 Messages
   Description                                                                          File     Line  Colu...  Project
1  The field 'WindowsFormsApplication11.Form1.riddleDoor1' is assigned but its value is never used   Game.cs  16    14       WindowsFormsApplication11
2  The field 'WindowsFormsApplication11.Form1.passDoor1' is assigned but its value is never used     Game.cs  17    14       WindowsFormsApplication11
3  The field 'WindowsFormsApplication11.Form1.collect32' is assigned but its value is never used     Game.cs  19    14       WindowsFormsApplication11
4  The field 'WindowsFormsApplication11.Form1.keyNo' is assigned but its value is never used         Game.cs  27    13       WindowsFormsApplic
Build succeeded
```

Figure 24: Changed TextBox output to inventory box

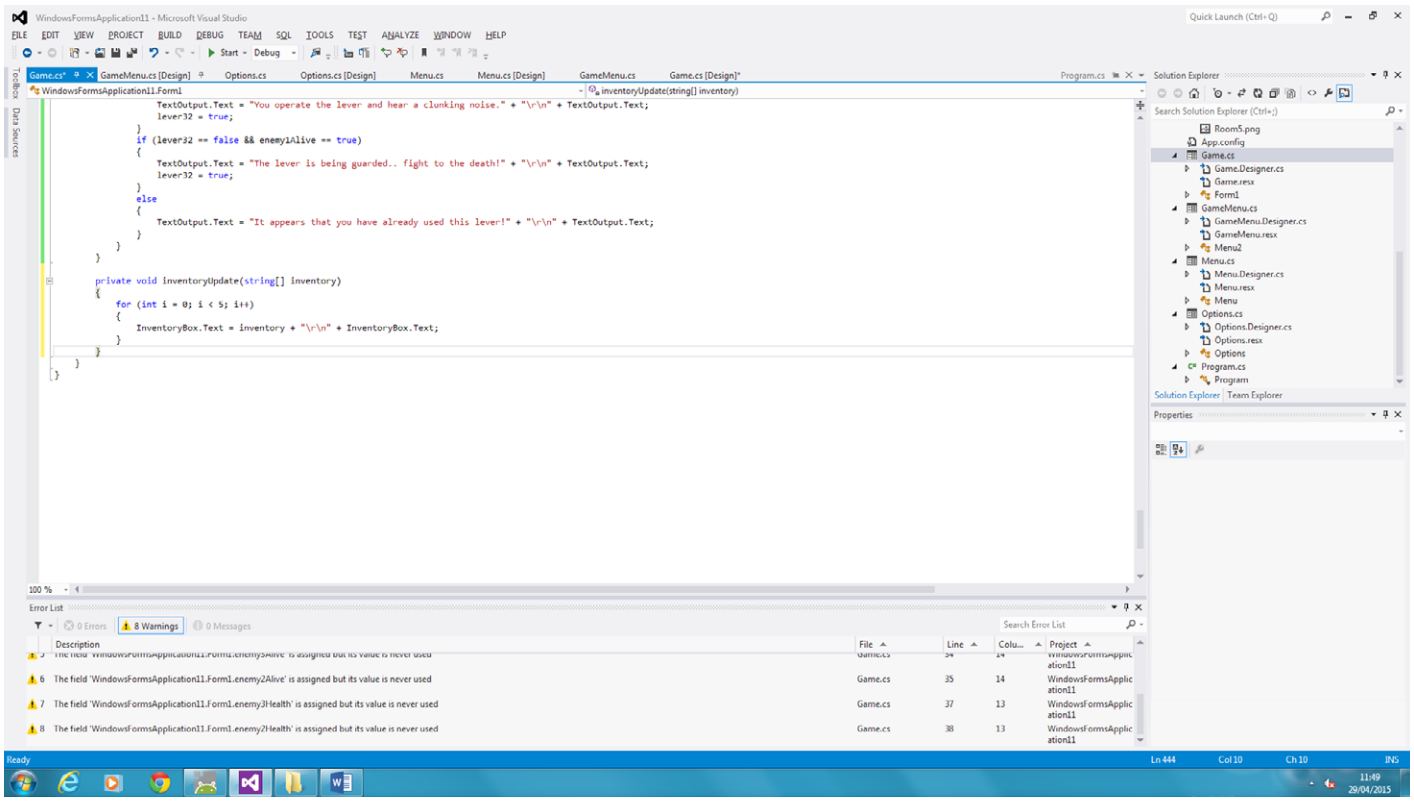


```
            if (activeMonsterHealth > 0)
            {
                // Player attacks monster
                TextOutput.Text = "Monster 1 has " + activeMonsterHealth + " hit points, you set a defensive stance" + "\r\n" + TextOutput.Text;
                activeMonsterHealth = (int)((double)activeMonsterHealth - (double)chance.Next(5, 10) * 0.5);
                //Monster attacks player
                TextOutput.Text = "You have " + playerHealth + " hit points.. the monster attacks you!" + "\r\n" + TextOutput.Text;

                block = chance.Next(4, 7) * chance.Next(0,1);
                playerHealth = playerHealth - block;
                if (block == 0)
                {
                    TextOutput.Text = "You have blocked all damage that the monster did to you!" + "\r\n" + TextOutput.Text;
                }
                else
                {
                    TextOutput.Text = "You have failed to block and the monster hit you for" + block + "hit points" + "\r\n" + TextOutput.Text;
                }
                //Output results
                TextOutput.Text = "You have " + playerHealth + " hitpoints left" + "\r\n" + TextOutput.Text;
                if (activeMonsterHealth <= 0)
                {
                    TextOutput.Text = "The mostner has died" + "\r\n" + TextOutput.Text;
                }
                else
                {
                    TextOutput.Text = "The mostner has " + activeMonsterHealth + " hitpoints left" + "\r\n" + TextOutput.Text;
                }
            }
            else if (playerHealth <= 0)
            {
                TextOutput.Text = "Hahahahha you died!" + "\r\n" + TextOutput.Text;
            }

            else if (activeMonsterHealth <= 0)
            {
                TextOutput.Text = "The mostner is already dead, you hack into it's squisdy dead body.... and get covered in goo!" + "\r\n" + TextOutput.Text;
            }

            return activeMonsterHealth;
```

```
Error List
0 Errors   8 Warnings   0 Messages
   Description                                                                          File     Line  Colu...  Project
6  The field 'WindowsFormsApplication11.Form1.enemy2Alive' is assigned but its value is never used   Game.cs  35    14       WindowsFormsApplication11
7  The field 'WindowsFormsApplication11.Form1.enemy3Health' is assigned but its value is never used  Game.cs  37    13       WindowsFormsApplication11
8  The field 'WindowsFormsApplication11.Form1.enemy2Health' is assigned but its value is never used  Game.cs  38    13       WindowsFormsApplication11
Ready
```

Figure 25: Coded Inventory update for this room

The inventory box must be updated to show the player current inventory at all times. As such I created a method called "inventoryUpdate" which is called whenever the player picks up an item. The method then resets the inventory box's text to include the most recent values within the inventory array. It uses a FOR loop to repeat a line showing a single inventory slot whether it is empty or not. This is repeated a set amount of times.

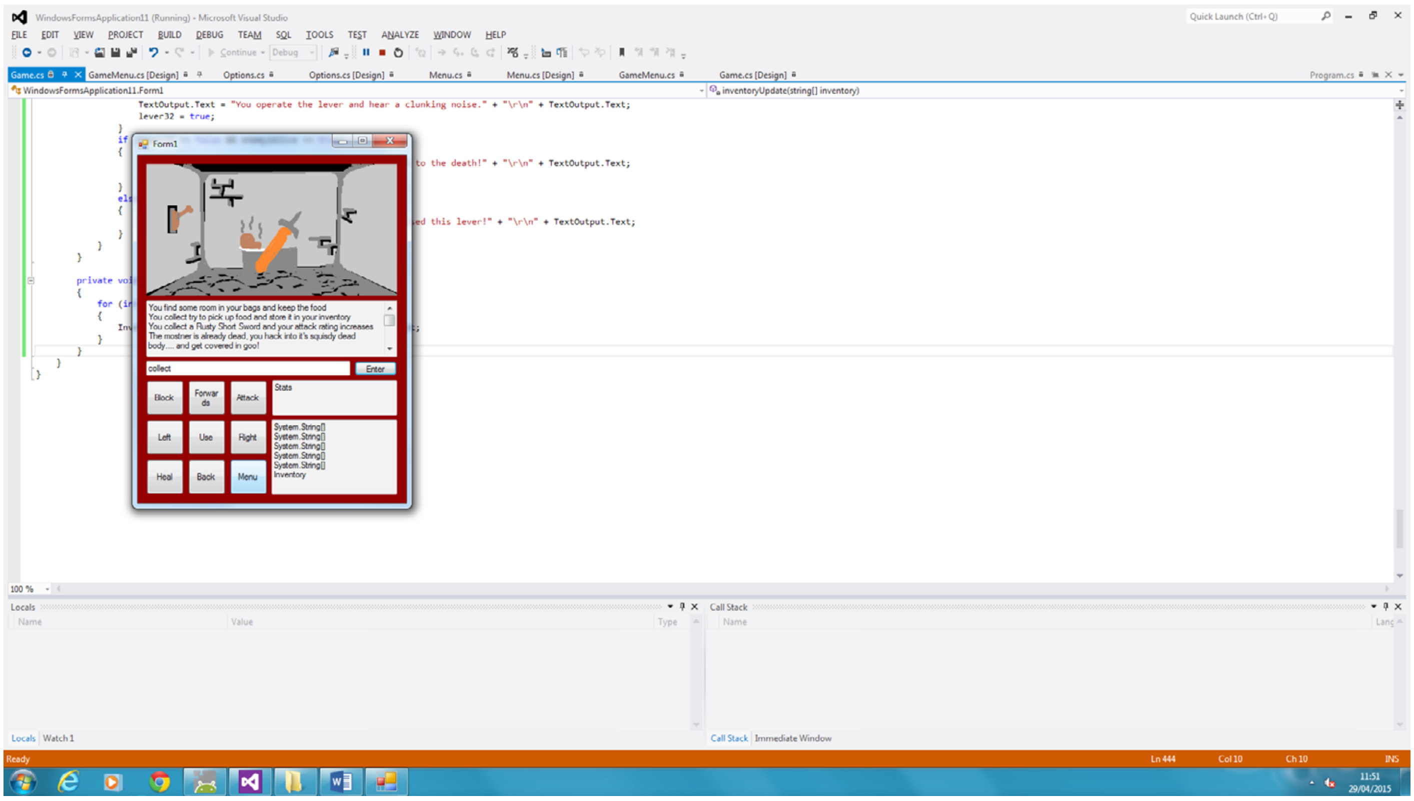


```
                TextOutput.Text = "You operate the lever and hear a clunking noise." + "\r\n" + TextOutput.Text;
                lever32 = true;
            }
            if (lever32 == false && enemy1Alive == true)
            {
                TextOutput.Text = "The lever is being guarded.. fight to the death!" + "\r\n" + TextOutput.Text;
                lever32 = true;
            }
            else
            {
                TextOutput.Text = "It appears that you have already used this lever!" + "\r\n" + TextOutput.Text;
            }
        }
    }

    private void inventoryUpdate(string[] inventory)
    {
        for (int i = 0; i < 5; i++)
        {
            InventoryBox.Text = inventory + "\r\n" + InventoryBox.Text;
        }
    }
  }
}
```

```
Error List    0 Errors    8 Warnings    0 Messages

   Description                                                                          File      Line  Colu...  Project
⚠ 6  The field 'WindowsFormsApplication11.Form1.enemy2Alive' is assigned but its value is never used    Game.cs   35    14   WindowsFormsApplication11
⚠ 7  The field 'WindowsFormsApplication11.Form1.enemy3Health' is assigned but its value is never used   Game.cs   37    13   WindowsFormsApplication11
⚠ 8  The field 'WindowsFormsApplication11.Form1.enemy2Health' is assigned but its value is never used   Game.cs   38    13   WindowsFormsApplication11
```

Figure 26: added inventory update method as this will be needed for other rooms.

While testing the program I ran Into the problem that the inventory was being displayed as "System.String[]" rather than "Item: Empty slot". This was due to the code printing inventory without accessing a specific area of the array. This was easily fixed by adding "[i]" to the end of inventory. This means that for every loop in the FOR loop, the next item will be accessed.



```
You find some room in your bags and keep the food
You collect try to pick up food and store it in your inventory
You collect a Rusty Short Sword and your attack rating increases
The mostner is already dead, you hack into it's squisdy dead
body..... and get covered in goo!
```

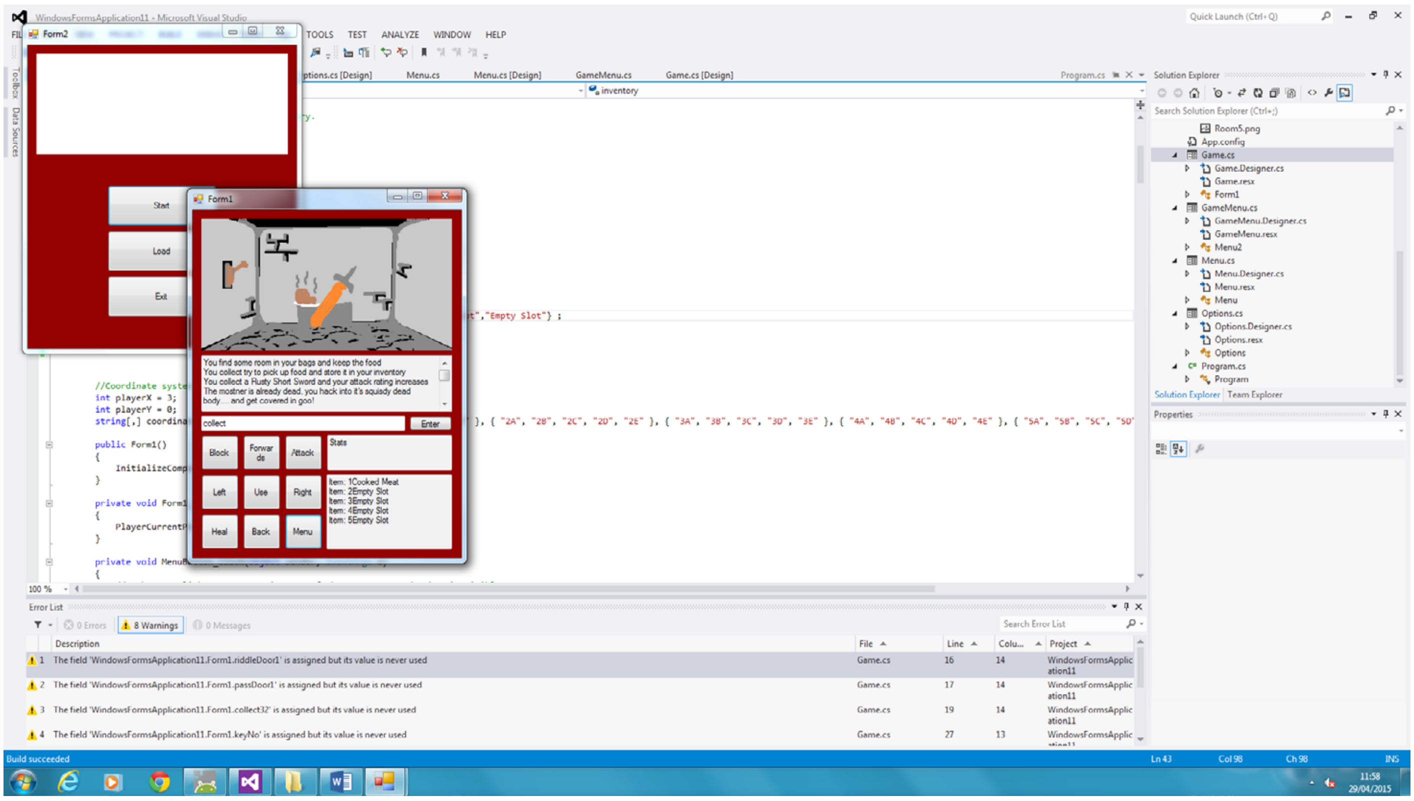
```
collect
```

```
Stats
```

```
System.String[]
System.String[]
System.String[]
System.String[]
System.String[]
Inventory
```

Figure 27: Works but inventory not printing accurately

Once this was fixed the code needed to be formatted so that there was a space between the item number and the item.



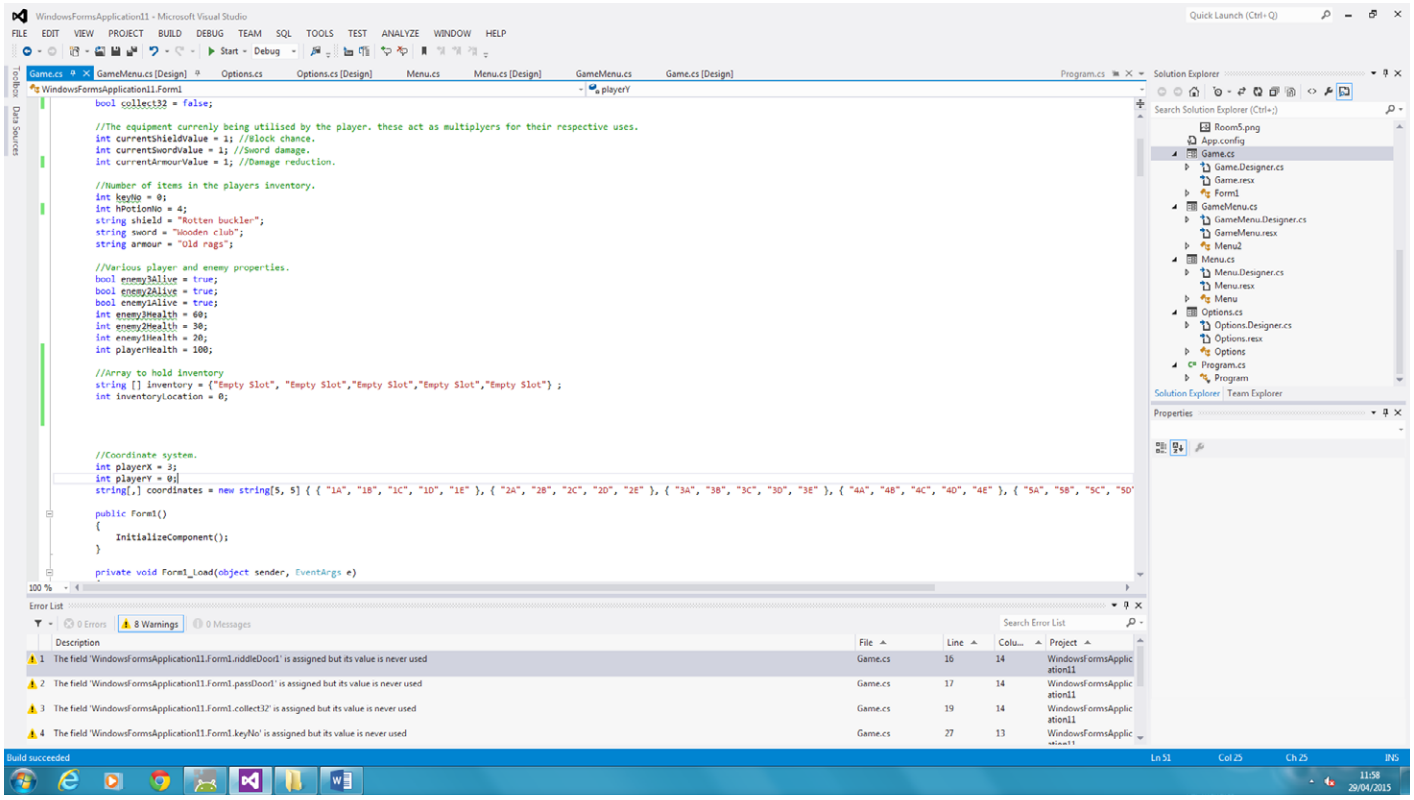
```
You find some room in your bags and keep the food
You collect try to pick up food and store it in your inventory
You collect a Rusty Short Sword and your attack rating increases
The mostner is already dead. you hack into it's squisdy dead
body.... and get covered in goo!
```

```
Item: 1Cooked Meat
Item: 2Empty Slot
Item: 3Empty Slot
Item: 4Empty Slot
Item: 5Empty Slot
```

	Description	File	Line	Colu...	Project
1	The field 'WindowsFormsApplication11.Form1.riddleDoor1' is assigned but its value is never used	Game.cs	16	14	WindowsFormsApplication11
2	The field 'WindowsFormsApplication11.Form1.passDoor1' is assigned but its value is never used	Game.cs	17	14	WindowsFormsApplication11
3	The field 'WindowsFormsApplication11.Form1.collect32' is assigned but its value is never used	Game.cs	19	14	WindowsFormsApplication11
4	The field 'WindowsFormsApplication11.Form1.keyNo' is assigned but its value is never used	Game.cs	27	13	WindowsFormsApplic

Figure 28: Working inventory – needs formatting

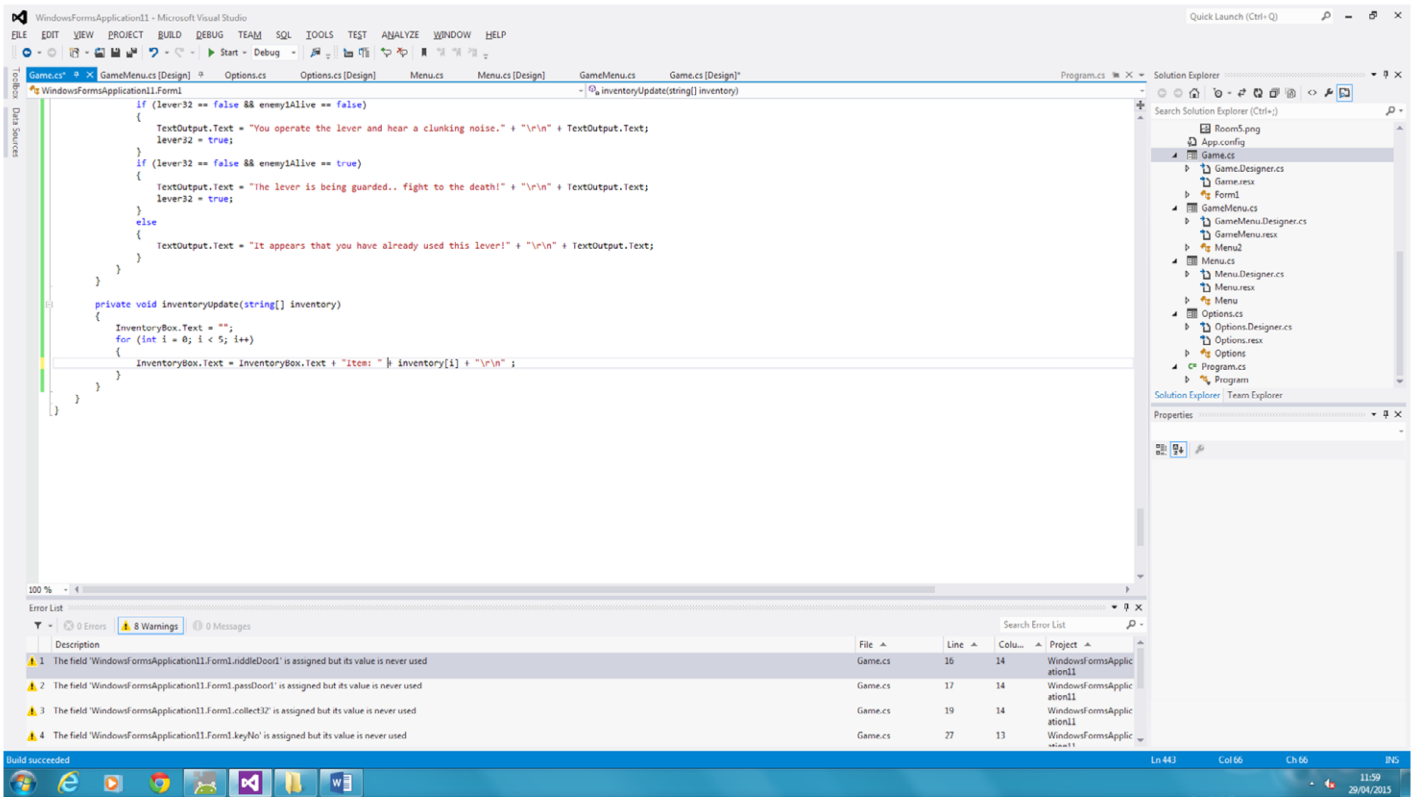
Where the inventory array is declared, the string "Empty slot" is preloaded into the array to fill the vacant spaces in the players inventory.



```
bool collect32 = false;

//The equipment currenly being utilised by the player. these act as multiplyers for their respective uses.
int currentShieldValue = 1; //Block chance.
int currentSwordValue = 1; //Sword damage.
int currentArmourValue = 1; //Damage reduction.

//Number of items in the players inventory.
int keyNo = 0;
int hPotionNo = 4;
string shield = "Rotten buckler";
string sword = "Wooden club";
string armour = "Old rags";

//Various player and enemy properties.
bool enemy3Alive = true;
bool enemy2Alive = true;
bool enemy1Alive = true;
int enemy3Health = 60;
int enemy2Health = 30;
int enemy1Health = 20;
int playerHealth = 100;

//Array to hold inventory
string [] inventory = {"Empty Slot", "Empty Slot","Empty Slot","Empty Slot","Empty Slot"} ;
int inventoryLocation = 0;



//Coordinate system.
int playerX = 3;
int playerY = 0;
string[,] coordinates = new string[5, 5] { { "1A", "1B", "1C", "1D", "1E" }, { "2A", "2B", "2C", "2D", "2E" }, { "3A", "3B", "3C", "3D", "3E" }, { "4A", "4B", "4C", "4D", "4E" }, { "5A", "5B", "5C", "5D"

public Form1()
{
    InitializeComponent();
}

private void Form1_Load(object sender, EventArgs e)
```

	Description	File	Line	Colu...	Project
1	The field 'WindowsFormsApplication11.Form1.riddleDoor1' is assigned but its value is never used	Game.cs	16	14	WindowsFormsApplication11
2	The field 'WindowsFormsApplication11.Form1.passDoor1' is assigned but its value is never used	Game.cs	17	14	WindowsFormsApplication11
3	The field 'WindowsFormsApplication11.Form1.collect32' is assigned but its value is never used	Game.cs	19	14	WindowsFormsApplication11
4	The field 'WindowsFormsApplication11.Form1.keyNo' is assigned but its value is never used	Game.cs	27	13	WindowsFormsApplic

Figure 29: Preloaded "Empty Slot"



```
            if (lever32 == false && enemy1Alive == false)
            {
                TextOutput.Text = "You operate the lever and hear a clunking noise." + "\r\n" + TextOutput.Text;
                lever32 = true;
            }
            if (lever32 == false && enemy1Alive == true)
            {
                TextOutput.Text = "The lever is being guarded.. fight to the death!" + "\r\n" + TextOutput.Text;
                lever32 = true;
            }
            else
            {
                TextOutput.Text = "It appears that you have already used this lever!" + "\r\n" + TextOutput.Text;
            }
        }
    }

    private void inventoryUpdate(string[] inventory)
    {
        InventoryBox.Text = "";
        for (int i = 0; i < 5; i++)
        {
            InventoryBox.Text = InventoryBox.Text + "Item: " + inventory[i] + "\r\n" ;
        }
    }
  }
}
```

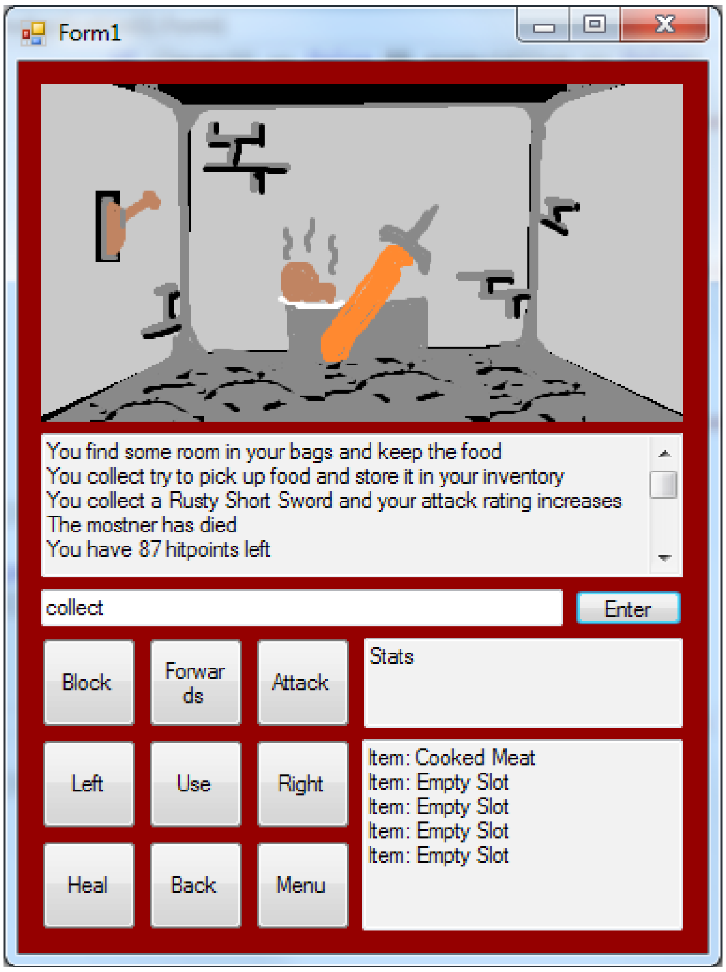
Error List: 0 Errors, 8 Warnings, 0 Messages

	Description	File	Line	Colu...	Project
1	The field 'WindowsFormsApplication11.Form1.riddleDoor1' is assigned but its value is never used	Game.cs	16	14	WindowsFormsApplication11
2	The field 'WindowsFormsApplication11.Form1.passDoor1' is assigned but its value is never used	Game.cs	17	14	WindowsFormsApplication11
3	The field 'WindowsFormsApplication11.Form1.collect32' is assigned but its value is never used	Game.cs	19	14	WindowsFormsApplication11
4	The field 'WindowsFormsApplication11.Form1.keyNo' is assigned but its value is never used	Game.cs	27	13	WindowsFormsApplication11

Build succeeded

Figure 30: Edited Inventory Function

Once the changes had been made to the inventory system I ran a test. The system worked as intended.



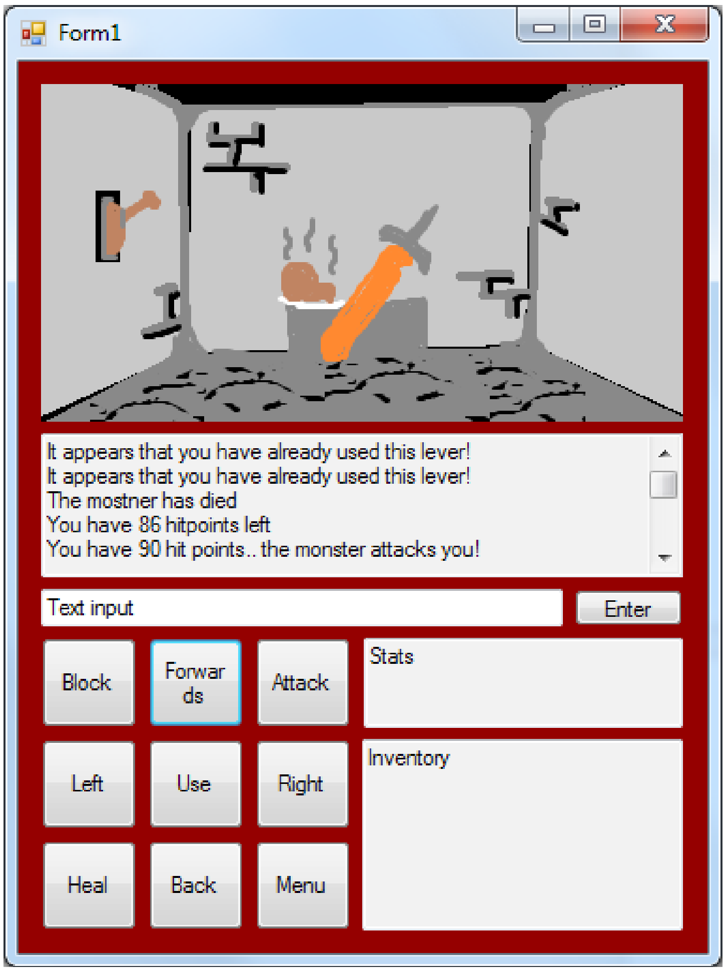
```
You find some room in your bags and keep the food
You collect try to pick up food and store it in your inventory
You collect a Rusty Short Sword and your attack rating increases
The mostner has died
You have 87 hitpoints left
```

collect

```
Item: Cooked Meat
Item: Empty Slot
Item: Empty Slot
Item: Empty Slot
Item: Empty Slot
```

Figure 31: Working Inventory system

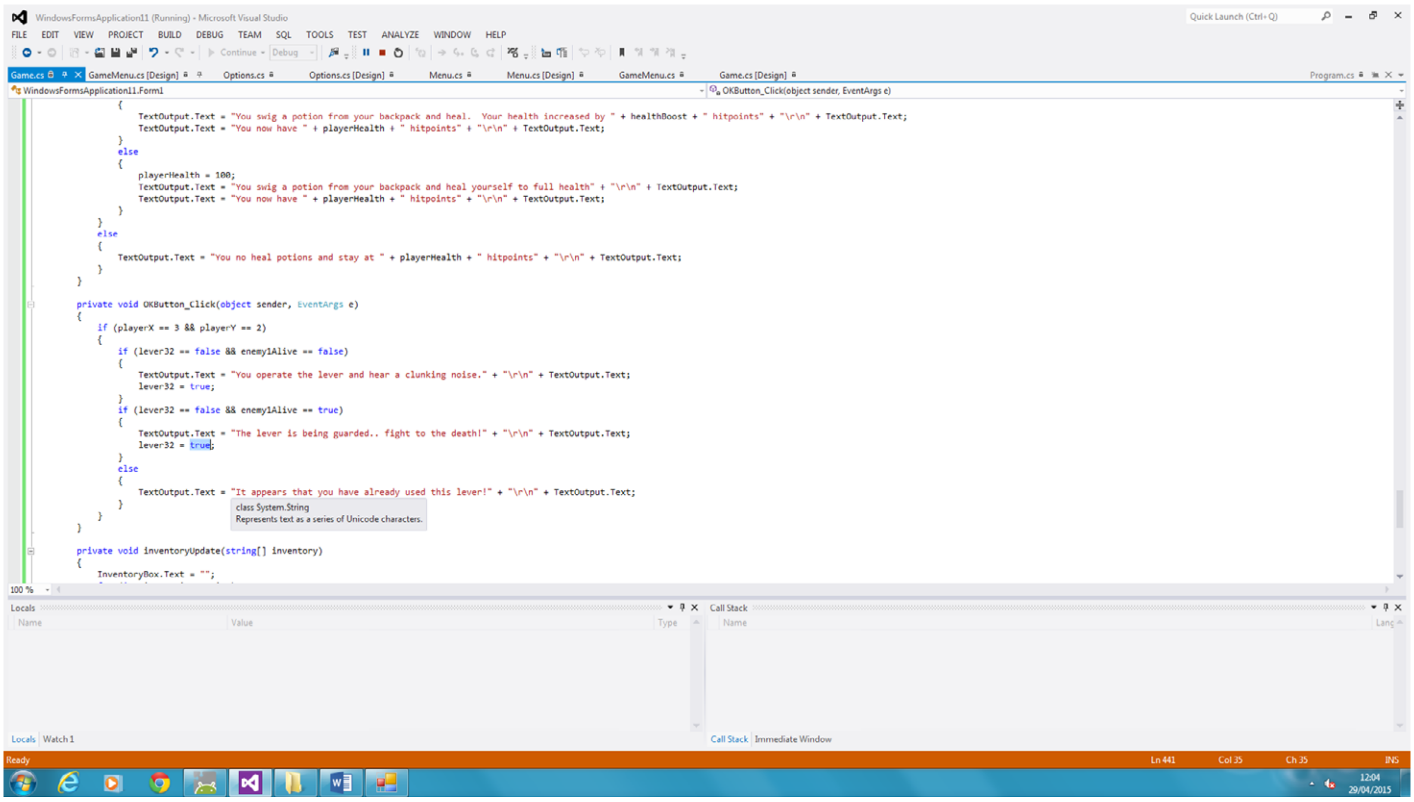
During the test I realised that the lever in room (3, 2) was being activated when the player "used" it even when the monster was still alive, the player was also still informed that the enemy was blocking the lever. This was due to both IF statements containing "lever32 = true". I fixed this problem by removing this line of code from the IF statement containing "enemy1Alive == true".



```
It appears that you have already used this lever!
It appears that you have already used this lever!
The mostner has died
You have 86 hitpoints left
You have 90 hit points.. the monster attacks you!
```

Text input

Figure 32: lever operated even when monster is alive

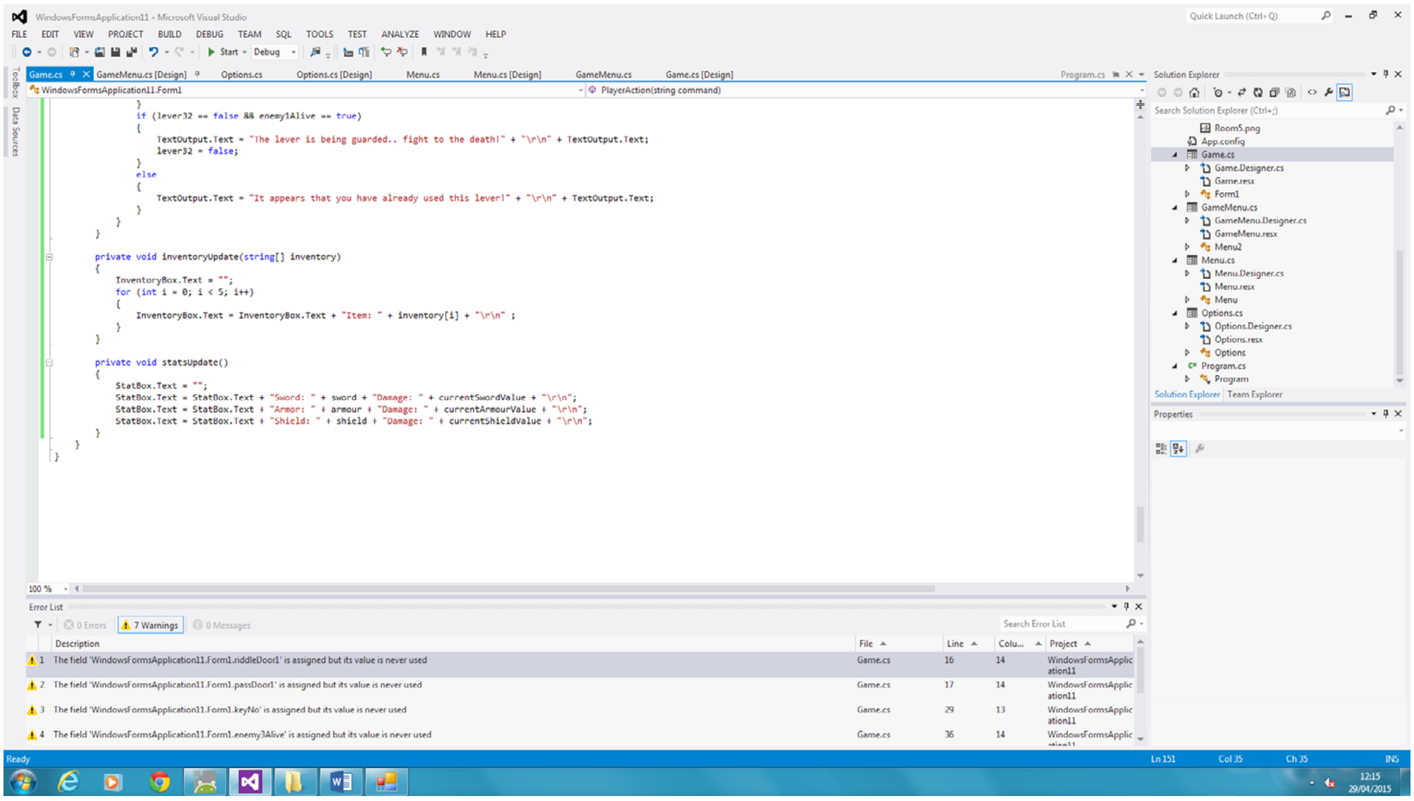


```
            {
                TextOutput.Text = "You swig a potion from your backpack and heal.  Your health increased by " + healthBoost + " hitpoints" + "\r\n" + TextOutput.Text;
                TextOutput.Text = "You now have " + playerHealth + " hitpoints" + "\r\n" + TextOutput.Text;
            }
            else
            {
                playerHealth = 100;
                TextOutput.Text = "You swig a potion from your backpack and heal yourself to full health" + "\r\n" + TextOutput.Text;
                TextOutput.Text = "You now have " + playerHealth + " hitpoints" + "\r\n" + TextOutput.Text;
            }
        }
        else
        {
            TextOutput.Text = "You no heal potions and stay at " + playerHealth + " hitpoints" + "\r\n" + TextOutput.Text;
        }
    }

    private void OKButton_Click(object sender, EventArgs e)
    {
        if (playerX == 3 && playerY == 2)
        {
            if (lever32 == false && enemy1Alive == false)
            {
                TextOutput.Text = "You operate the lever and hear a clunking noise." + "\r\n" + TextOutput.Text;
                lever32 = true;
            }
            if (lever32 == false && enemy1Alive == true)
            {
                TextOutput.Text = "The lever is being guarded.. fight to the death!" + "\r\n" + TextOutput.Text;
                lever32 = true;
            }
            else
            {
                TextOutput.Text = "It appears that you have already used this lever!" + "\r\n" + TextOutput.Text;
            }
        }
    }

    private void inventoryUpdate(string[] inventory)
    {
        InventoryBox.Text = "";
```

class System.String
Represents text as a series of Unicode characters.

Figure 33: Corrective Action – set to false

Similar to the inventory box, the stat box needs to be updated whenever a change is made to any of the values related to the stat box. As such I created a method called "statsUpdate" which is called whenever the player's equipment is changed. It clears the stat box and replaces the contents with up to date values.



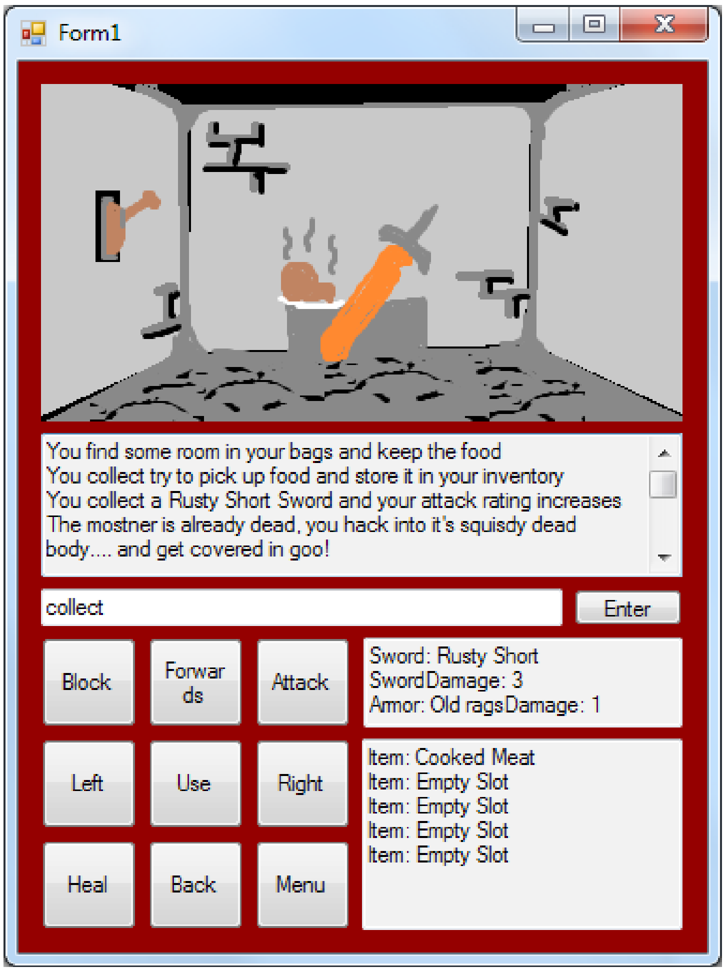
```
                }
                if (lever32 == false && enemy1Alive == true)
                {
                    TextOutput.Text = "The lever is being guarded.. fight to the death!" + "\r\n" + TextOutput.Text;
                    lever32 = false;
                }
                else
                {
                    TextOutput.Text = "It appears that you have already used this lever!" + "\r\n" + TextOutput.Text;
                }
            }
        }

        private void inventoryUpdate(string[] inventory)
        {
            InventoryBox.Text = "";
            for (int i = 0; i < 5; i++)
            {
                InventoryBox.Text = InventoryBox.Text + "Item: " + inventory[i] + "\r\n" ;
            }
        }

        private void statsUpdate()
        {
            StatBox.Text = "";
            StatBox.Text = StatBox.Text + "Sword: " + sword + "Damage: " + currentSwordValue + "\r\n";
            StatBox.Text = StatBox.Text + "Armor: " + armour + "Damage: " + currentArmourValue + "\r\n";
            StatBox.Text = StatBox.Text + "Shield: " + shield + "Damage: " + currentShieldValue + "\r\n";
        }
    }
}
```

Error List — 0 Errors, 7 Warnings, 0 Messages

	Description	File	Line	Colu...	Project
1	The field 'WindowsFormsApplication11.Form1.riddleDoor1' is assigned but its value is never used	Game.cs	16	14	WindowsFormsApplication11
2	The field 'WindowsFormsApplication11.Form1.passDoor1' is assigned but its value is never used	Game.cs	17	14	WindowsFormsApplication11
3	The field 'WindowsFormsApplication11.Form1.keyNo' is assigned but its value is never used	Game.cs	29	13	WindowsFormsApplication11
4	The field 'WindowsFormsApplication11.Form1.enemy3Alive' is assigned but its value is never used	Game.cs	36	14	WindowsFormsApplic

Figure 34: Added Stats Update method

A test showed that the stat box worked as it should however there was a minor formatting problem that was easily remedied. The box was too small to fit the name of the starter sword and its damage. This was fixed by shortening "Damage", "Defence" and "Block" to "Dmg", "Def" and "Blk".



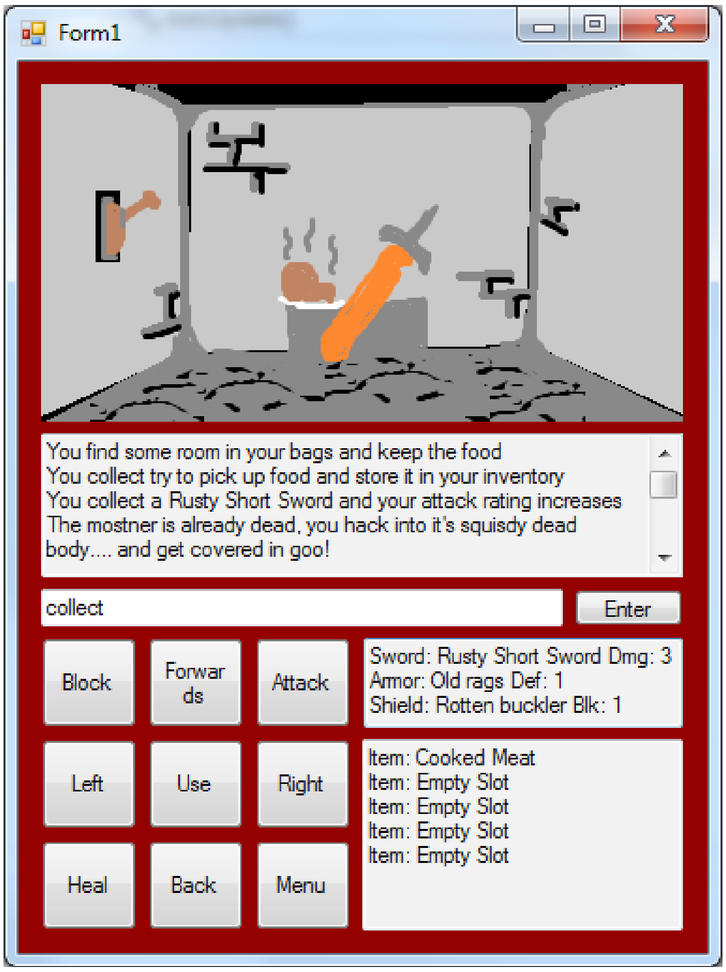
```
You find some room in your bags and keep the food
You collect try to pick up food and store it in your inventory
You collect a Rusty Short Sword and your attack rating increases
The mostner is already dead, you hack into it's squisdy dead
body.... and get covered in goo!
```

```
collect
```

```
Sword: Rusty Short
SwordDamage: 3
Armor: Old ragsDamage: 1
```

```
Item: Cooked Meat
Item: Empty Slot
Item: Empty Slot
Item: Empty Slot
Item: Empty Slot
```

Figure 35: StatsUpdate works but formatting issues (box too small)



```
You find some room in your bags and keep the food
You collect try to pick up food and store it in your inventory
You collect a Rusty Short Sword and your attack rating increases
The mostner is already dead, you hack into it's squisdy dead
body.... and get covered in goo!
```

```
collect
```

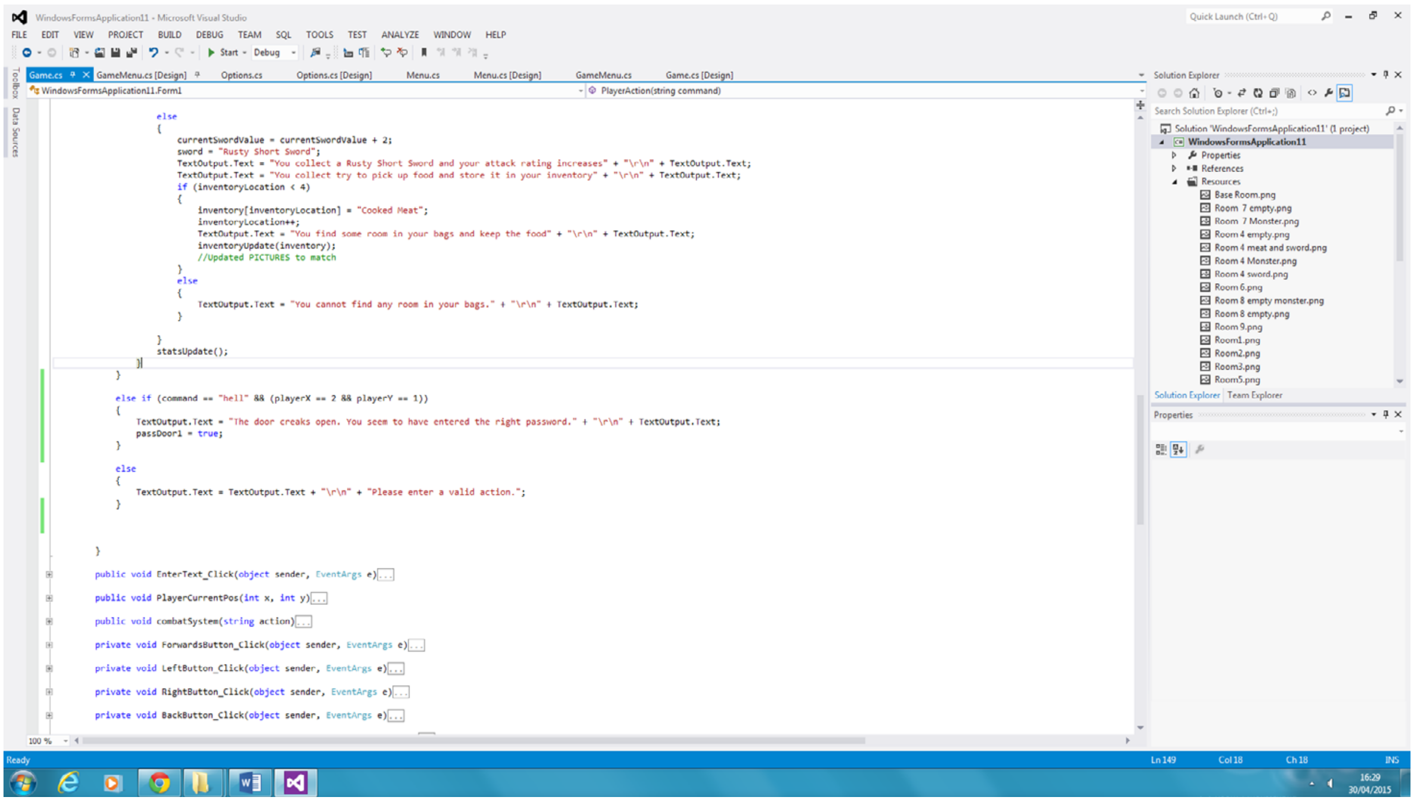
```
Sword: Rusty Short Sword Dmg: 3
Armor: Old rags Def: 1
Shield: Rotten buckler Blk: 1
```

```
Item: Cooked Meat
Item: Empty Slot
Item: Empty Slot
Item: Empty Slot
Item: Empty Slot
```

Figure 36: edited formatting for Stats Box

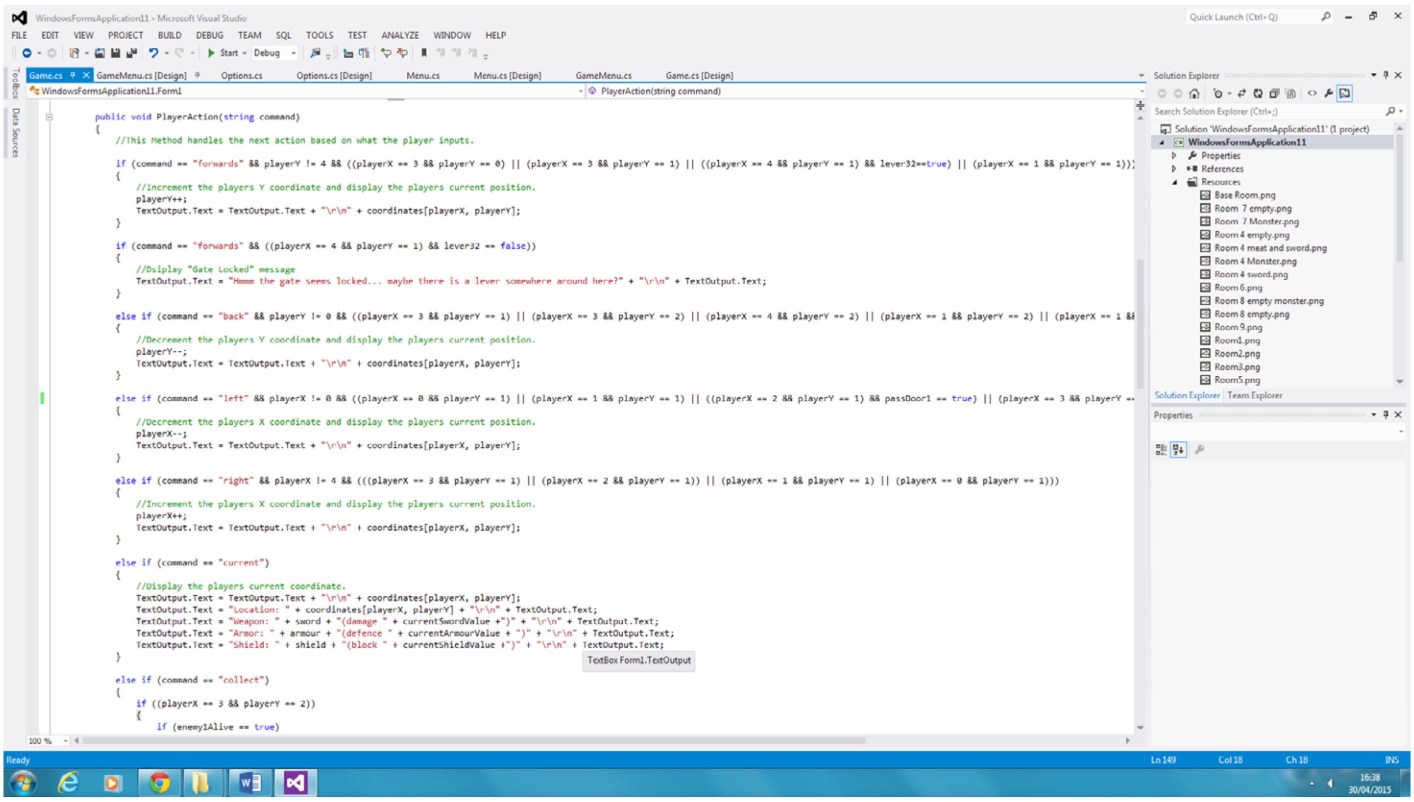
Now that each of the main systems are in a working state and are ready to be applied to the rest of the program I have decided to work on the riddle, password and locked doors. I will start with the password door.

I created a new IF statement to the "playerAction" method that runs when the player enters the command "hell" and is in room (2, 1). It informs the player that the door has opened and sets the variable "passDoor1" to true. I added a new logic statement to the "left" command IF statement so that the player may only go left in room (2, 1) when "passDoor1" is true.



```
                else
                {
                    currentSwordValue = currentSwordValue + 2;
                    sword = "Rusty Short Sword";
                    TextOutput.Text = "You collect a Rusty Short Sword and your attack rating increases" + "\r\n" + TextOutput.Text;
                    TextOutput.Text = "You collect try to pick up food and store it in your inventory" + "\r\n" + TextOutput.Text;
                    if (inventoryLocation < 4)
                    {
                        inventory[inventoryLocation] = "Cooked Meat";
                        inventoryLocation++;
                        TextOutput.Text = "You find some room in your bags and keep the food" + "\r\n" + TextOutput.Text;
                        inventoryUpdate(inventory);
                        //Updated PICTURES to match
                    }
                    else
                    {
                        TextOutput.Text = "You cannot find any room in your bags." + "\r\n" + TextOutput.Text;
                    }

                }
                statsUpdate();
            }
        }

        else if (command == "hell" && (playerX == 2 && playerY == 1))
        {
            TextOutput.Text = "The door creaks open. You seem to have entered the right password." + "\r\n" + TextOutput.Text;
            passDoor1 = true;
        }

        else
        {
            TextOutput.Text = TextOutput.Text + "\r\n" + "Please enter a valid action.";
        }


    }

    public void EnterText_Click(object sender, EventArgs e)...

    public void PlayerCurrentPos(int x, int y)...

    public void combatSystem(string action)...

    private void ForwardsButton_Click(object sender, EventArgs e)...

    private void LeftButton_Click(object sender, EventArgs e)...

    private void RightButton_Click(object sender, EventArgs e)...

    private void BackButton_Click(object sender, EventArgs e)...
```

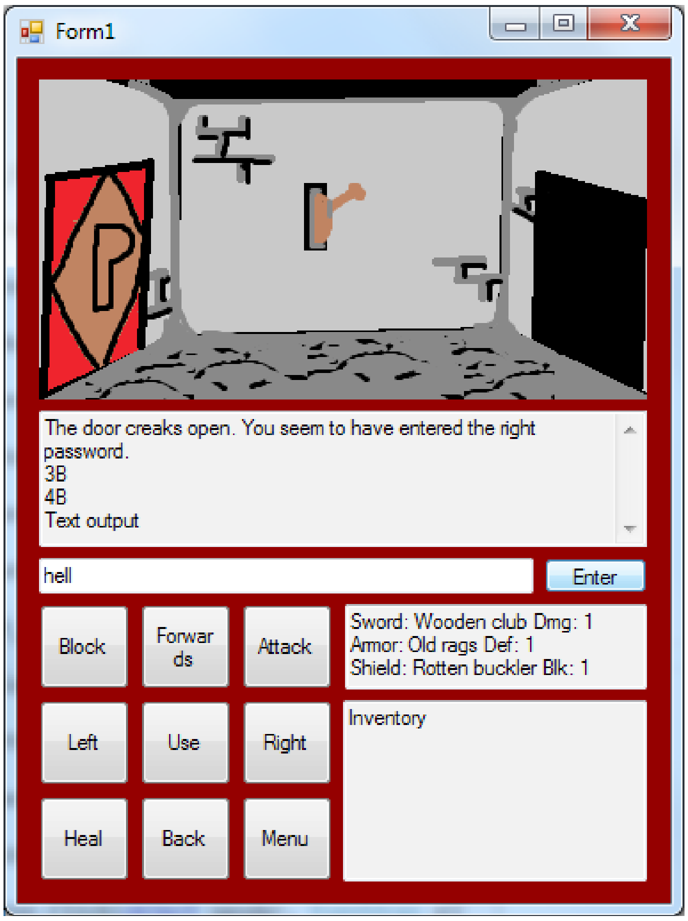
Figure 37: Added IF statement for "hell" command.



```
    public void PlayerAction(string command)
    {
        //This Method handles the next action based on what the player inputs.

        if (command == "forwards" && playerY != 4 && ((playerX == 3 && playerY == 0) || (playerX == 3 && playerY == 1) || ((playerX == 4 && playerY == 1) && lever32==true) || (playerX == 1 && playerY == 1));
        {
            //Increment the players Y coordinate and display the players current position.
            playerY++;
            TextOutput.Text = TextOutput.Text + "\r\n" + coordinates[playerX, playerY];
        }

        if (command == "forwards" && ((playerX == 4 && playerY == 1) && lever32 == false))
        {
            //Dsiplay "Gate Locked" message
            TextOutput.Text = "Hmmm the gate seems locked... maybe there is a lever somewhere around here?" + "\r\n" + TextOutput.Text;
        }

        else if (command == "back" && playerY != 0 && ((playerX == 3 && playerY == 1) || (playerX == 3 && playerY == 2) || (playerX == 4 && playerY == 2) || (playerX == 1 && playerY == 2) || (playerX == 1 &&
        {
            //Decrement the players Y coordinate and display the players current position.
            playerY--;
            TextOutput.Text = TextOutput.Text + "\r\n" + coordinates[playerX, playerY];
        }

        else if (command == "left" && playerX != 0 && ((playerX == 0 && playerY == 1) || (playerX == 1 && playerY == 1) || ((playerX == 2 && playerY == 1) && passDoor1 == true) || (playerX == 3 && playerY ==
        {
            //Decrement the players X coordinate and display the players current position.
            playerX--;
            TextOutput.Text = TextOutput.Text + "\r\n" + coordinates[playerX, playerY];
        }

        else if (command == "right" && playerX != 4 && (((playerX == 3 && playerY == 1) || (playerX == 2 && playerY == 1)) || (playerX == 1 && playerY == 1) || (playerX == 0 && playerY == 1)))
        {
            //Increment the players X coordinate and display the players current position.
            playerX++;
            TextOutput.Text = TextOutput.Text + "\r\n" + coordinates[playerX, playerY];
        }

        else if (command == "current")
        {
            //Display the players current coordinate.
            TextOutput.Text = TextOutput.Text + "\r\n" + coordinates[playerX, playerY];
            TextOutput.Text = "Location: " + coordinates[playerX, playerY] + "\r\n" + TextOutput.Text;
            TextOutput.Text = "Weapon: " + sword + "(damage " + currentSwordValue +")" + "\r\n" + TextOutput.Text;
            TextOutput.Text = "Armor: " + armour + "(defence " + currentArmourValue + ")" + "\r\n" + TextOutput.Text;
            TextOutput.Text = "Shield: " + shield + "(block " + currentShieldValue +")" + "\r\n" + TextOutput.Text;
        }

        else if (command == "collect")
        {
            if ((playerX == 3 && playerY == 2))
            {
                if (enemy1Alive == true)
```

Figure 38: Added logic statement to "left" command for room (2, 1).

As the code for this part was finished I decided to test it. The door worked as it should. The next door I will create will be the riddle door. The riddle door will work exactly the same as the password door except text will be output beforehand to inform the player of the riddle.



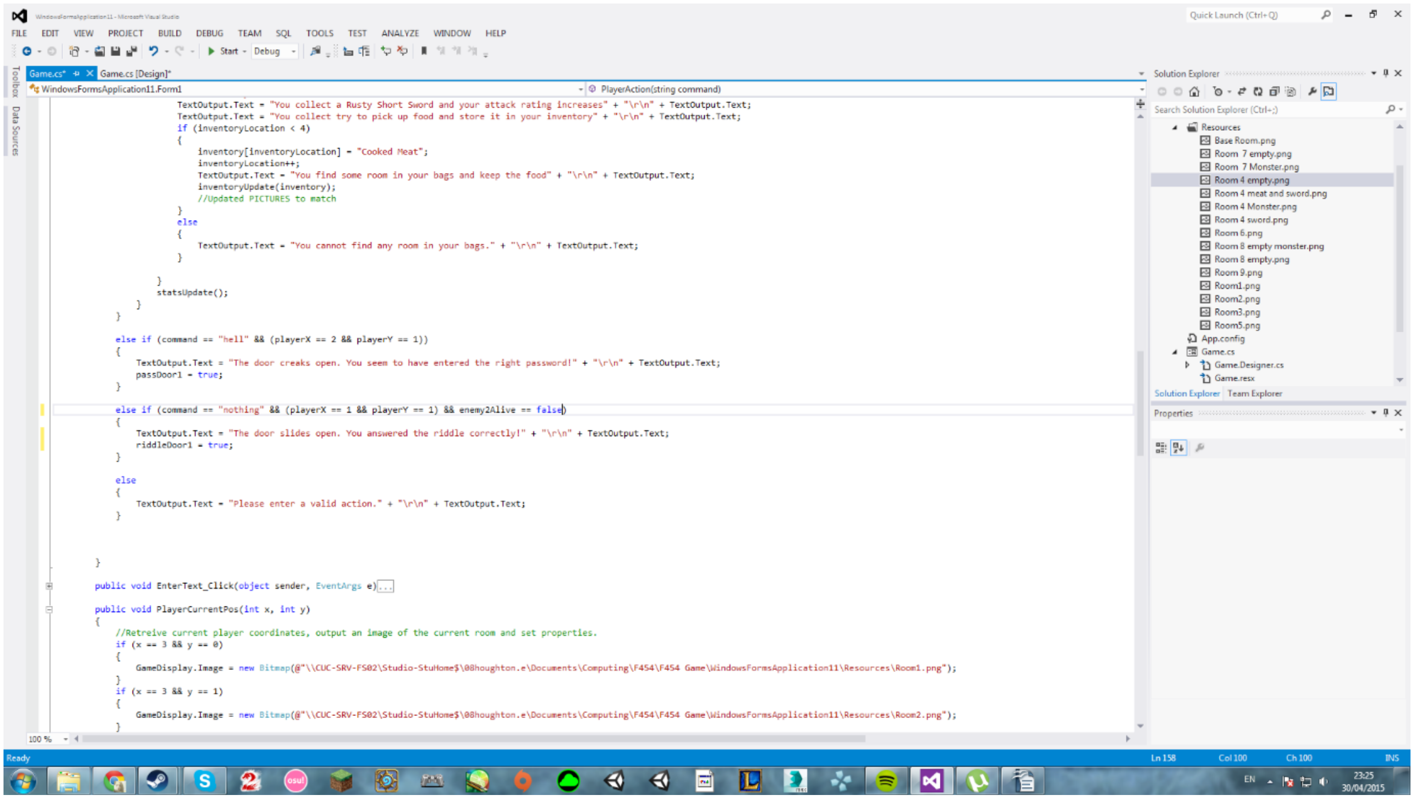
```
The door creaks open. You seem to have entered the right
password.
3B
4B
Text output
```

hell

Sword: Wooden club Dmg: 1
Armor: Old rags Def: 1
Shield: Rotten buckler Blk: 1

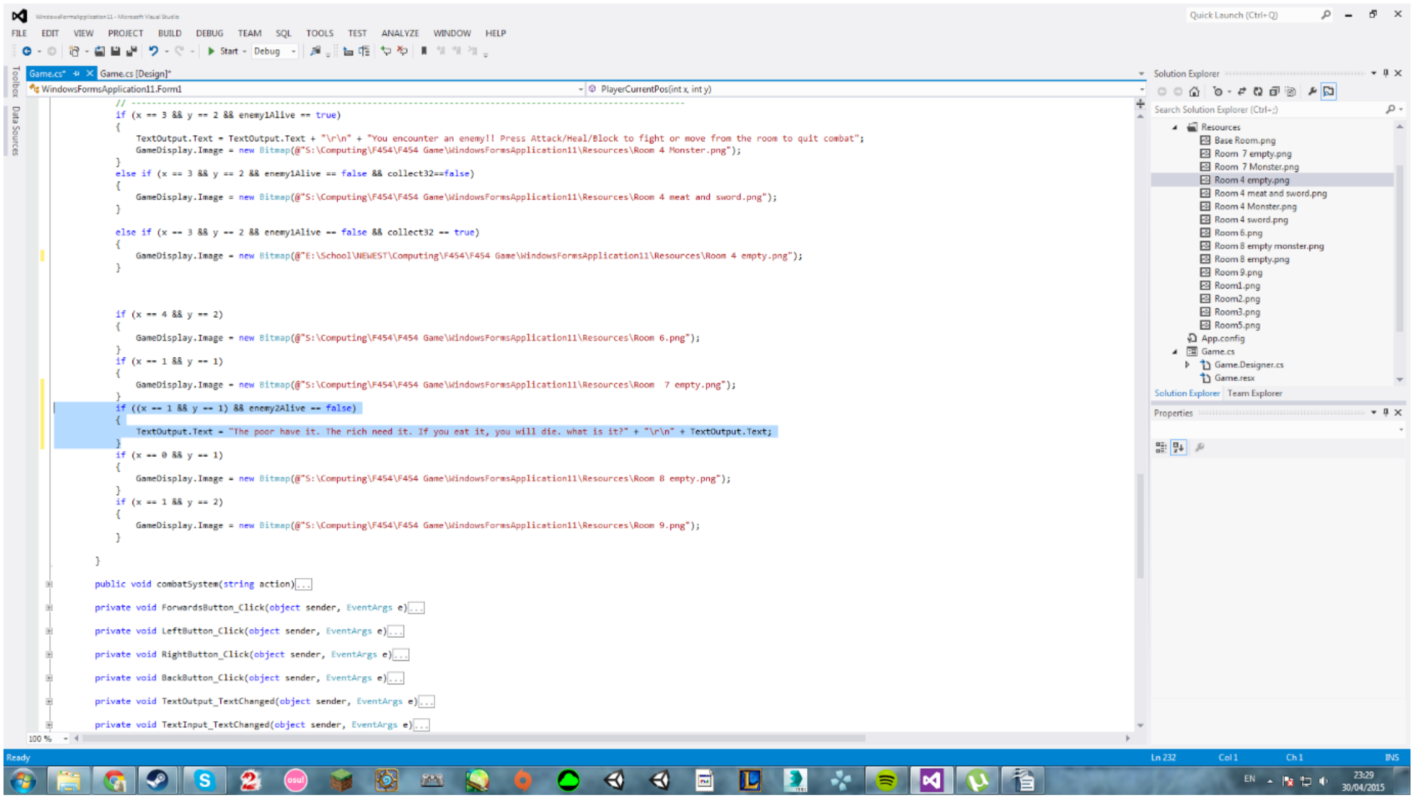
Figure 39: Password door work as intended. No problems.

I followed the same steps as when I was creating the password door code. When the player enters the command "nothing" in room (1, 1) once the monster is defeated, the door will be unlocked and the player can proceed. However as this is a riddle door the riddle must be shown in the output box whenever the player enters the room and the enemy is dead.



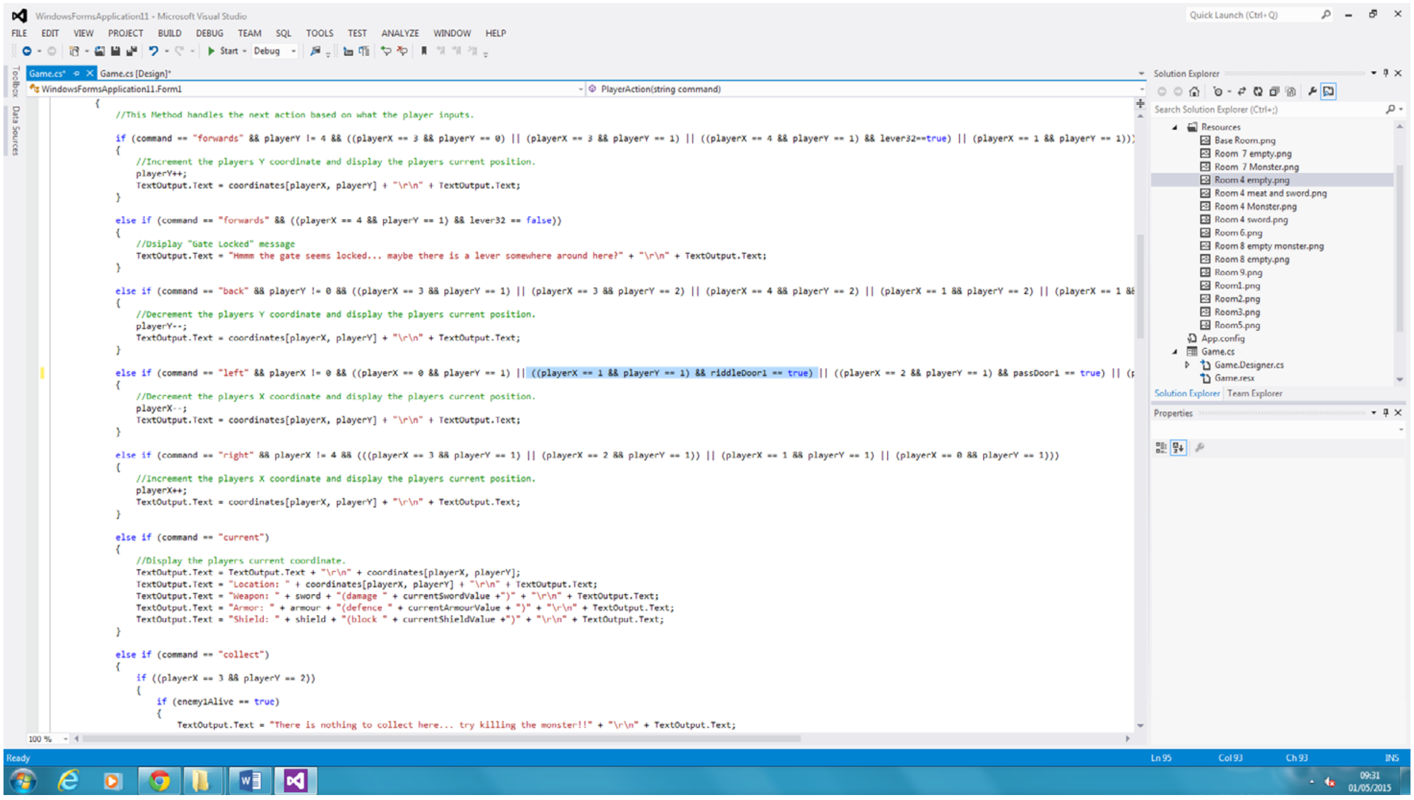
```
                TextOutput.Text = "You collect a Rusty Short Sword and your attack rating increases" + "\r\n" + TextOutput.Text;
                TextOutput.Text = "You collect try to pick up food and store it in your inventory" + "\r\n" + TextOutput.Text;
                if (inventoryLocation < 4)
                {
                    inventory[inventoryLocation] = "Cooked Meat";
                    inventoryLocation++;
                    TextOutput.Text = "You find some room in your bags and keep the food" + "\r\n" + TextOutput.Text;
                    inventoryUpdate(inventory);
                    //Updated PICTURES to match
                }
                else
                {
                    TextOutput.Text = "You cannot find any room in your bags." + "\r\n" + TextOutput.Text;
                }

            }
            statsUpdate();
        }
    }

    else if (command == "hell" && (playerX == 2 && playerY == 1))
    {
        TextOutput.Text = "The door creaks open. You seem to have entered the right password!" + "\r\n" + TextOutput.Text;
        passDoor1 = true;
    }

    else if (command == "nothing" && (playerX == 1 && playerY == 1) && enemy2Alive == false)
    {
        TextOutput.Text = "The door slides open. You answered the riddle correctly!" + "\r\n" + TextOutput.Text;
        riddleDoor1 = true;
    }

    else
    {
        TextOutput.Text = "Please enter a valid action." + "\r\n" + TextOutput.Text;
    }


}

public void EnterText_Click(object sender, EventArgs e)...

public void PlayerCurrentPos(int x, int y)
{
    //Retreive current player coordinates, output an image of the current room and set properties.
    if (x == 3 && y == 0)
    {
        GameDisplay.Image = new Bitmap(@"\\CUC-SRV-FS02\Studio-StuHome$\08houghton.e\Documents\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room1.png");
    }
    if (x == 3 && y == 1)
    {
        GameDisplay.Image = new Bitmap(@"\\CUC-SRV-FS02\Studio-StuHome$\08houghton.e\Documents\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room2.png");
    }
```

Figure 40: Created "nothing" command.



```
            // ------------------------------------------------------------------------------------------------
            if (x == 3 && y == 2 && enemy1Alive == true)
            {
                TextOutput.Text = TextOutput.Text + "\r\n" + "You encounter an enemy!! Press Attack/Heal/Block to fight or move from the room to quit combat";
                GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room 4 Monster.png");
            }
            else if (x == 3 && y == 2 && enemy1Alive == false && collect32==false)
            {
                GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room 4 meat and sword.png");
            }

            else if (x == 3 && y == 2 && enemy1Alive == false && collect32 == true)
            {
                GameDisplay.Image = new Bitmap(@"E:\School\NEWEST\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room 4 empty.png");
            }



            if (x == 4 && y == 2)
            {
                GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room 6.png");
            }
            if (x == 1 && y == 1)
            {
                GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room  7 empty.png");
            }
            if ((x == 1 && y == 1) && enemy2Alive == false)
            {
                TextOutput.Text = "The poor have it. The rich need it. If you eat it, you will die. what is it?" + "\r\n" + TextOutput.Text;
            }
            if (x == 0 && y == 1)
            {
                GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room 8 empty.png");
            }
            if (x == 1 && y == 2)
            {
                GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room 9.png");
            }

        }

        public void combatSystem(string action)...

        private void ForwardsButton_Click(object sender, EventArgs e)...

        private void LeftButton_Click(object sender, EventArgs e)...

        private void RightButton_Click(object sender, EventArgs e)...

        private void BackButton_Click(object sender, EventArgs e)...

        private void TextOutput_TextChanged(object sender, EventArgs e)...

        private void TextInput_TextChanged(object sender, EventArgs e)...
```

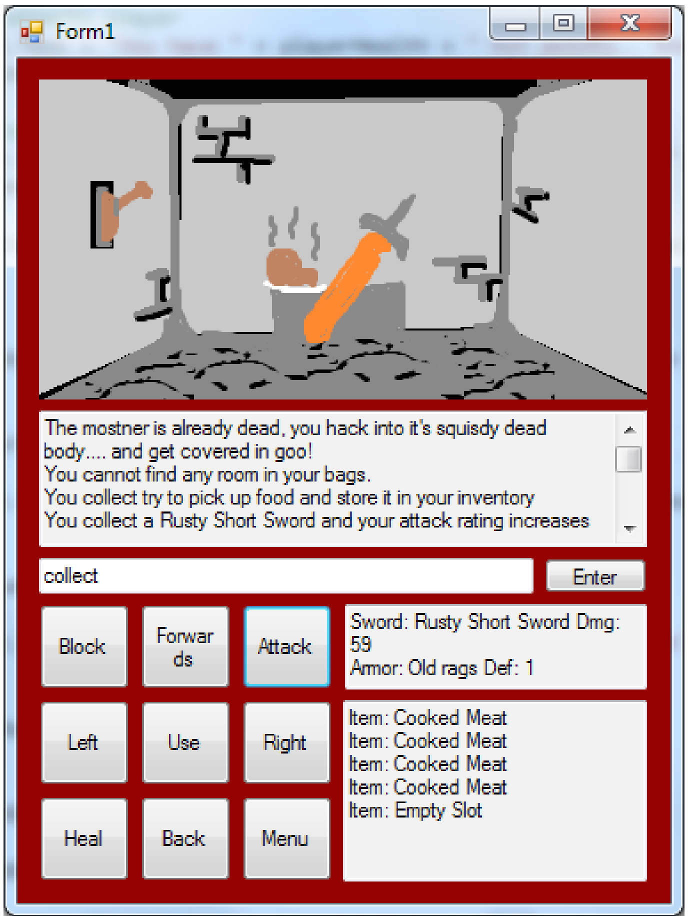
Figure 41: Text output box shows riddle in text output upon entering room (1, 1).



```
        {
            //This Method handles the next action based on what the player inputs.

            if (command == "forwards" && playerY != 4 && ((playerX == 3 && playerY == 0) || (playerX == 3 && playerY == 1) || ((playerX == 4 && playerY == 1) && lever32==true) || (playerX == 1 && playerY == 1)))
            {
                //Increment the players Y coordinate and display the players current position.
                playerY++;
                TextOutput.Text = coordinates[playerX, playerY] + "\r\n" + TextOutput.Text;
            }

            else if (command == "forwards" && ((playerX == 4 && playerY == 1) && lever32 == false))
            {
                //Dsiplay "Gate Locked" message
                TextOutput.Text = "Hmmm the gate seems locked... maybe there is a lever somewhere around here?" + "\r\n" + TextOutput.Text;
            }

            else if (command == "back" && playerY != 0 && ((playerX == 3 && playerY == 1) || (playerX == 3 && playerY == 2) || (playerX == 4 && playerY == 2) || (playerX == 1 && playerY == 2) || (playerX == 1 &&
            {
                //Decrement the players Y coordinate and display the players current position.
                playerY--;
                TextOutput.Text = coordinates[playerX, playerY] + "\r\n" + TextOutput.Text;
            }

            else if (command == "left" && playerX != 0 && ((playerX == 0 && playerY == 1) || ((playerX == 1 && playerY == 1) && riddleDoor1 == true) || ((playerX == 2 && playerY == 1) && passDoor1 == true) || (
            {
                //Decrement the players X coordinate and display the players current position.
                playerX--;
                TextOutput.Text = coordinates[playerX, playerY] + "\r\n" + TextOutput.Text;
            }

            else if (command == "right" && playerX != 4 && (((playerX == 3 && playerY == 1) || (playerX == 2 && playerY == 1)) || (playerX == 1 && playerY == 1) || (playerX == 0 && playerY == 1)))
            {
                //Increment the players X coordinate and display the players current position.
                playerX++;
                TextOutput.Text = coordinates[playerX, playerY] + "\r\n" + TextOutput.Text;
            }

            else if (command == "current")
            {
                //Display the players current coordinate.
                TextOutput.Text = TextOutput.Text + "\r\n" + coordinates[playerX, playerY];
                TextOutput.Text = "Location: " + coordinates[playerX, playerY] + "\r\n" + TextOutput.Text;
                TextOutput.Text = "Weapon: " + sword + "(damage " + currentSwordValue +")" + "\r\n" + TextOutput.Text;
                TextOutput.Text = "Armor: " + armour + "(defence " + currentArmourValue + ")" + "\r\n" + TextOutput.Text;
                TextOutput.Text = "Shield: " + shield + "(block " + currentShieldValue +")" + "\r\n" + TextOutput.Text;
            }

            else if (command == "collect")
            {
                if ((playerX == 3 && playerY == 2))
                {
                    if (enemy1Alive == true)
                    {
                        TextOutput.Text = "There is nothing to collect here... try killing the monster!!" + "\r\n" + TextOutput.Text;
```

Figure 42: Added logic to "left" command for room (1, 1) riddle door.

I ran a test to see if the riddle door was working. The door worked as it should however if the player entered the command "nothing" after the door had already been opened, the "door has been opened" message would appear again. I also noticed a few problems in room (3, 2). Once the player had picked up the items in the room, the image would not change to the empty room. If the player "collected" the items more than once, the swords damage would keep increasing. There were also some spelling errors in the text.



```
The mostner is already dead, you hack into it's squisdy dead
body.... and get covered in goo!
You cannot find any room in your bags.
You collect try to pick up food and store it in your inventory
You collect a Rusty Short Sword and your attack rating increases
```

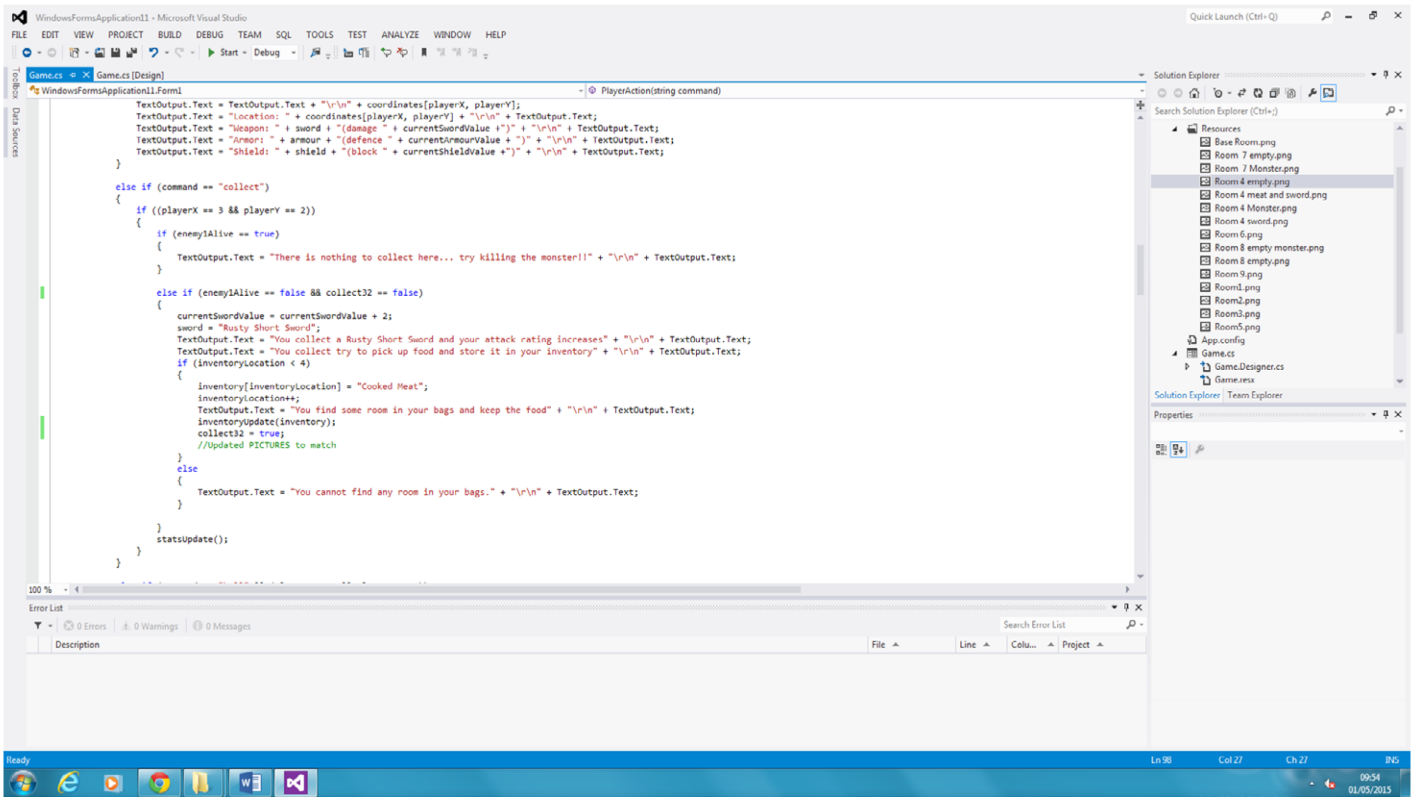
```
Sword: Rusty Short Sword Dmg:
59
Armor: Old rags Def: 1
```

```
Item: Cooked Meat
Item: Cooked Meat
Item: Cooked Meat
Item: Cooked Meat
Item: Empty Slot
```

Figure 43: Image not changing to empty room on item pickup, increasing damage and spelling errors.

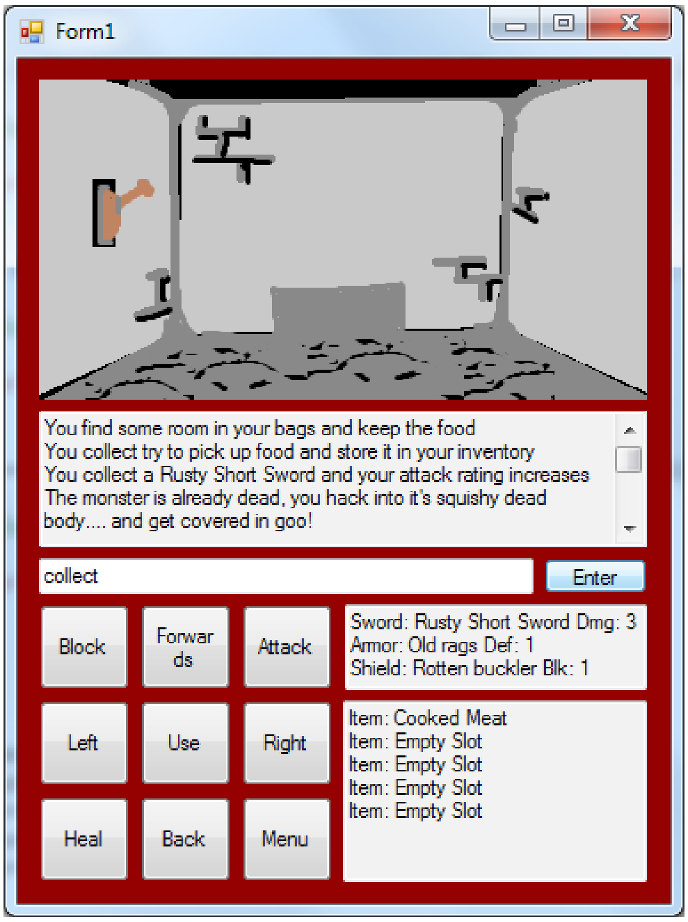
I fixed the increasing damage and image problems by changing the ELSE statement in the "collect" command code to an ELSE IF that runs if the enemy is dead and the items haven't been collected yet. At the end of the statement the variable "collect32" is set to true so that the statement will not run again. This also allows the room image to change. I also corrected the spelling errors when the player attacks a dead enemy.

Once those were fixed I added a logic statement to the "nothing" command so that the statement will only run if the door has not already been opened which fixed the problem of repeating text.



```
            TextOutput.Text = TextOutput.Text + "\r\n" + coordinates[playerX, playerY];
            TextOutput.Text = "Location: " + coordinates[playerX, playerY] + "\r\n" + TextOutput.Text;
            TextOutput.Text = "Weapon: " + sword + "(damage " + currentSwordValue +")" + "\r\n" + TextOutput.Text;
            TextOutput.Text = "Armor: " + armour + "(defence " + currentArmourValue + ")" + "\r\n" + TextOutput.Text;
            TextOutput.Text = "Shield: " + shield + "(block " + currentShieldValue +")" + "\r\n" + TextOutput.Text;
        }

        else if (command == "collect")
        {
            if ((playerX == 3 && playerY == 2))
            {
                if (enemy1Alive == true)
                {
                    TextOutput.Text = "There is nothing to collect here... try killing the monster!!" + "\r\n" + TextOutput.Text;
                }

                else if (enemy1Alive == false && collect32 == false)
                {
                    currentSwordValue = currentSwordValue + 2;
                    sword = "Rusty Short Sword";
                    TextOutput.Text = "You collect a Rusty Short Sword and your attack rating increases" + "\r\n" + TextOutput.Text;
                    TextOutput.Text = "You collect try to pick up food and store it in your inventory" + "\r\n" + TextOutput.Text;
                    if (inventoryLocation < 4)
                    {
                        inventory[inventoryLocation] = "Cooked Meat";
                        inventoryLocation++;
                        TextOutput.Text = "You find some room in your bags and keep the food" + "\r\n" + TextOutput.Text;
                        inventoryUpdate(inventory);
                        collect32 = true;
                        //Updated PICTURES to match
                    }
                    else
                    {
                        TextOutput.Text = "You cannot find any room in your bags." + "\r\n" + TextOutput.Text;
                    }

                }
                statsUpdate();
            }
        }
```

Figure 44: Fixed image, spelling and "collect" code.



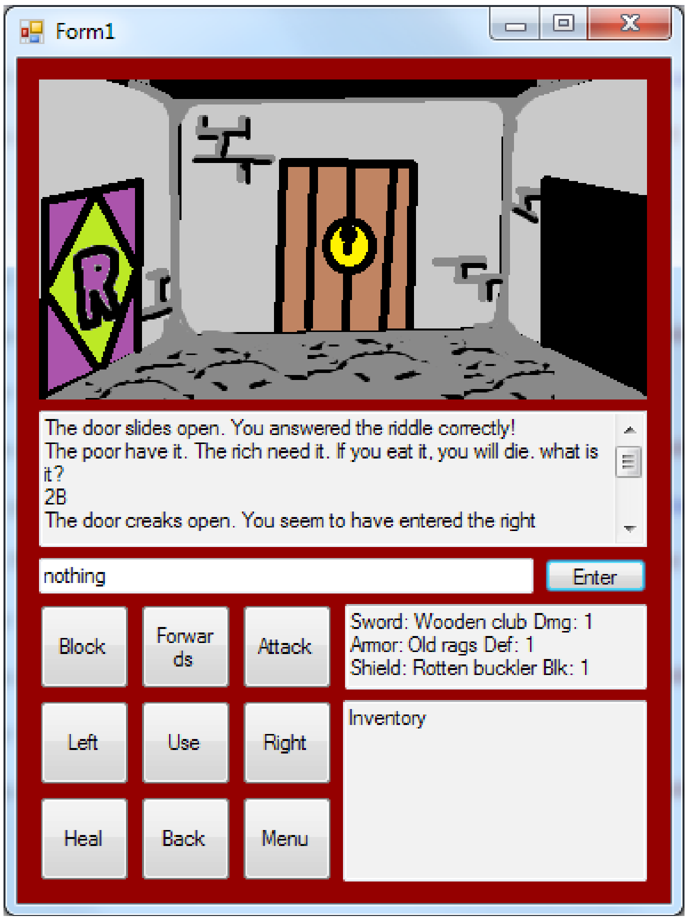
```
You find some room in your bags and keep the food
You collect try to pick up food and store it in your inventory
You collect a Rusty Short Sword and your attack rating increases
The monster is already dead, you hack into it's squishy dead
body.... and get covered in goo!
```

collect

```
Sword: Rusty Short Sword Dmg: 3
Armor: Old rags Def: 1
Shield: Rotten buckler Blk: 1
```

```
Item: Cooked Meat
Item: Empty Slot
Item: Empty Slot
Item: Empty Slot
Item: Empty Slot
```

Figure 45: Working room (3, 2).



```
The door slides open. You answered the riddle correctly!
The poor have it. The rich need it. If you eat it, you will die. what is
it?
2B
The door creaks open. You seem to have entered the right
```

```
nothing
```

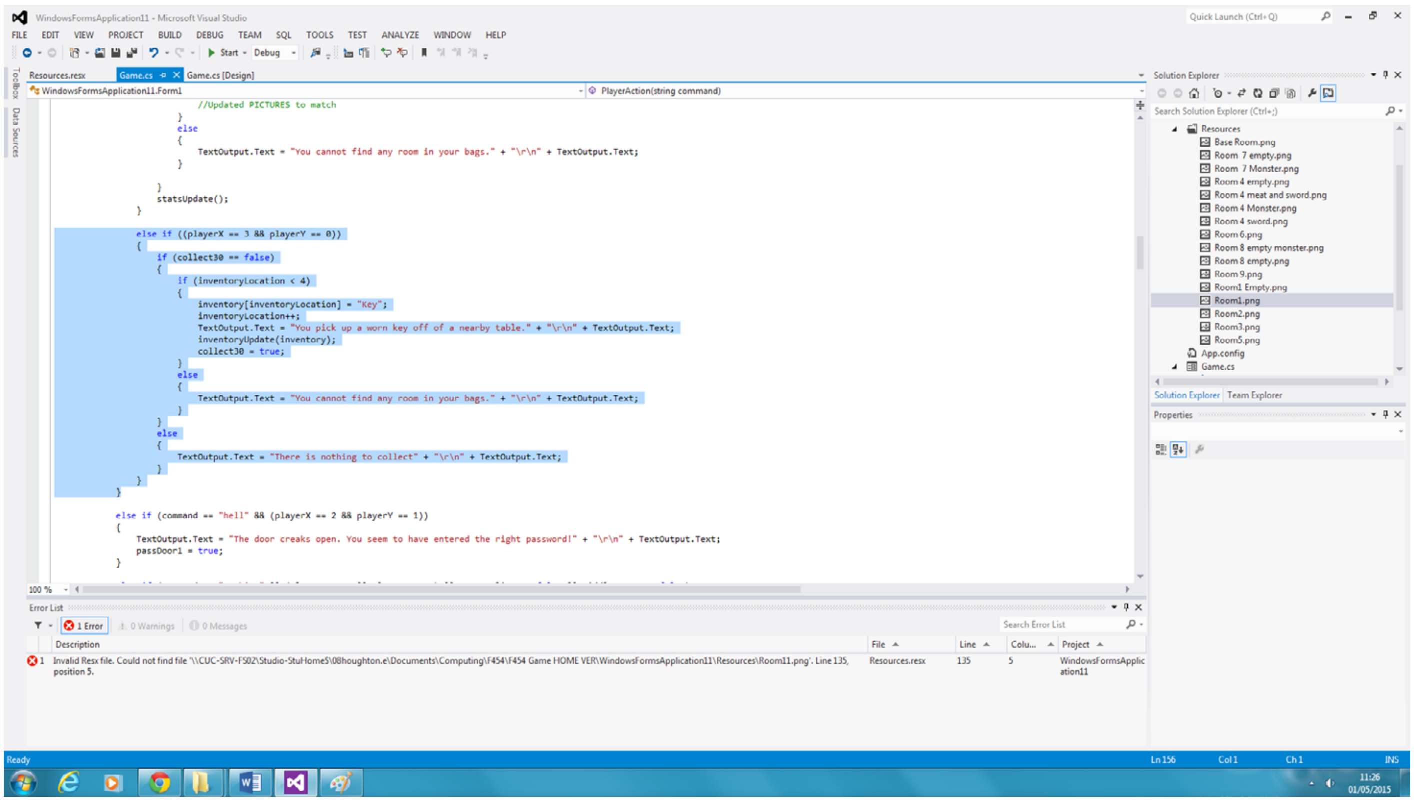
```
Sword: Wooden club Dmg: 1
Armor: Old rags Def: 1
Shield: Rotten buckler Blk: 1
```

Figure 46: Working riddle door.

The last system that needs to be developed is the locked door which will work differently to the other two doors in that in order to open the door, the player must be carrying a key in their inventory.

I started by making some code to let the player "collect" the key in the first room. I basically copied the code from room (3, 2)s collection but with some minor changes. The next empty inventory slot is retrieved and replaced with the string "Key".

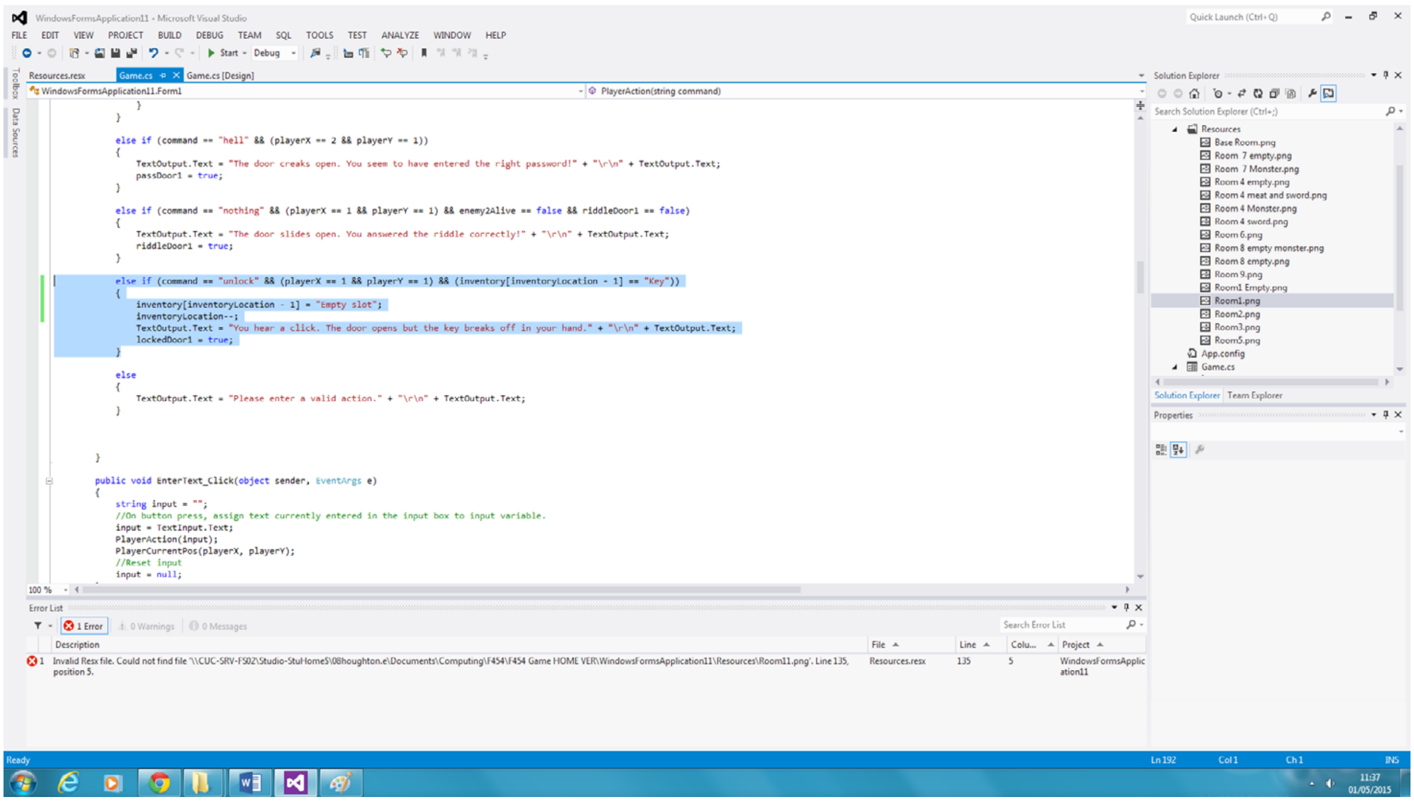
I created an ELSE IF statement within the "playerAction" method for the command "unlock" which opens the door if the player has a key present in their inventory. The key then breaks and is removed from the player's inventory.



```
                    //Updated PICTURES to match
                }
                else
                {
                    TextOutput.Text = "You cannot find any room in your bags." + "\r\n" + TextOutput.Text;
                }

            }
            statsUpdate();
        }

        else if ((playerX == 3 && playerY == 0))
        {
            if (collect30 == false)
            {
                if (inventoryLocation < 4)
                {
                    inventory[inventoryLocation] = "Key";
                    inventoryLocation++;
                    TextOutput.Text = "You pick up a worn key off of a nearby table." + "\r\n" + TextOutput.Text;
                    inventoryUpdate(inventory);
                    collect30 = true;
                }
                else
                {
                    TextOutput.Text = "You cannot find any room in your bags." + "\r\n" + TextOutput.Text;
                }
            }
            else
            {
                TextOutput.Text = "There is nothing to collect" + "\r\n" + TextOutput.Text;
            }
        }
    }

    else if (command == "hell" && (playerX == 2 && playerY == 1))
    {
        TextOutput.Text = "The door creaks open. You seem to have entered the right password!" + "\r\n" + TextOutput.Text;
        passDoor1 = true;
    }
```

```
1  Invalid Resx file. Could not find file '\\CUC-SRV-FS02\Studio-StuHomeS\08houghton.e\Documents\Computing\F454\F454 Game HOME VER\WindowsFormsApplication11\Resources\Room11.png'. Line 135, position 5.   Resources.resx   135   5   WindowsFormsApplication11
```

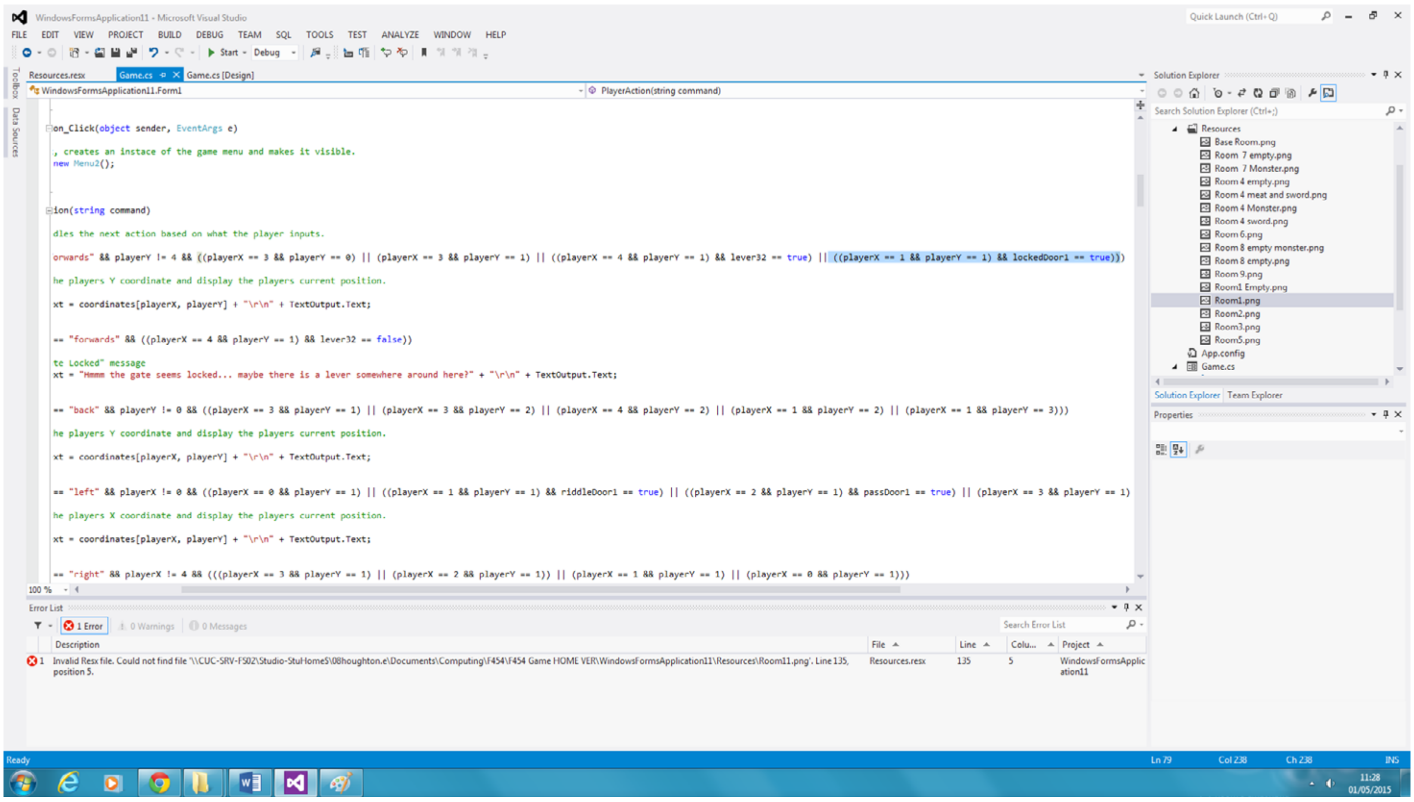
Figure 47: Created ELSE IF for key "collection".



```
            }
        }

        else if (command == "hell" && (playerX == 2 && playerY == 1))
        {
            TextOutput.Text = "The door creaks open. You seem to have entered the right password!" + "\r\n" + TextOutput.Text;
            passDoor1 = true;
        }

        else if (command == "nothing" && (playerX == 1 && playerY == 1) && enemy2Alive == false && riddleDoor1 == false)
        {
            TextOutput.Text = "The door slides open. You answered the riddle correctly!" + "\r\n" + TextOutput.Text;
            riddleDoor1 = true;
        }

        else if (command == "unlock" && (playerX == 1 && playerY == 1) && (inventory[inventoryLocation - 1] == "Key"))
        {
            inventory[inventoryLocation - 1] = "Empty slot";
            inventoryLocation--;
            TextOutput.Text = "You hear a click. The door opens but the key breaks off in your hand." + "\r\n" + TextOutput.Text;
            lockedDoor1 = true;
        }

        else
        {
            TextOutput.Text = "Please enter a valid action." + "\r\n" + TextOutput.Text;
        }



    }

    public void EnterText_Click(object sender, EventArgs e)
    {
        string input = "";
        //On button press, assign text currently entered in the input box to input variable.
        input = TextInput.Text;
        PlayerAction(input);
        PlayerCurrentPos(playerX, playerY);
        //Reset input
        input = null;
```

```
1  Invalid Resx file. Could not find file '\\CUC-SRV-FS02\Studio-StuHomeS\08houghton.e\Documents\Computing\F454\F454 Game HOME VER\WindowsFormsApplication11\Resources\Room11.png'. Line 135, position 5.   Resources.resx   135   5   WindowsFormsApplication11
```

Figure 48: Created "unlock" command.



```
on_Click(object sender, EventArgs e)

, creates an instace of the game menu and makes it visible.
new Menu2();


ion(string command)

dles the next action based on what the player inputs.

orwards" && playerY != 4 && ((playerX == 3 && playerY == 0) || (playerX == 3 && playerY == 1) || ((playerX == 4 && playerY == 1) && lever32 == true) || ((playerX == 1 && playerY == 1) && lockedDoor1 == true))

he players Y coordinate and display the players current position.

xt = coordinates[playerX, playerY] + "\r\n" + TextOutput.Text;


== "forwards" && ((playerX == 4 && playerY == 1) && lever32 == false))

te Locked" message
xt = "Hmmm the gate seems locked... maybe there is a lever somewhere around here?" + "\r\n" + TextOutput.Text;


== "back" && playerY != 0 && ((playerX == 3 && playerY == 1) || (playerX == 3 && playerY == 2) || (playerX == 4 && playerY == 2) || (playerX == 1 && playerY == 2) || (playerX == 1 && playerY == 3)))

he players Y coordinate and display the players current position.

xt = coordinates[playerX, playerY] + "\r\n" + TextOutput.Text;


== "left" && playerX != 0 && ((playerX == 0 && playerY == 1) || ((playerX == 1 && playerY == 1) && riddleDoor1 == true) || ((playerX == 2 && playerY == 1) && passDoor1 == true) || (playerX == 3 && playerY == 1)

he players X coordinate and display the players current position.

xt = coordinates[playerX, playerY] + "\r\n" + TextOutput.Text;


== "right" && playerX != 4 && (((playerX == 3 && playerY == 1) || (playerX == 2 && playerY == 1)) || (playerX == 1 && playerY == 1) || (playerX == 0 && playerY == 1)))
```

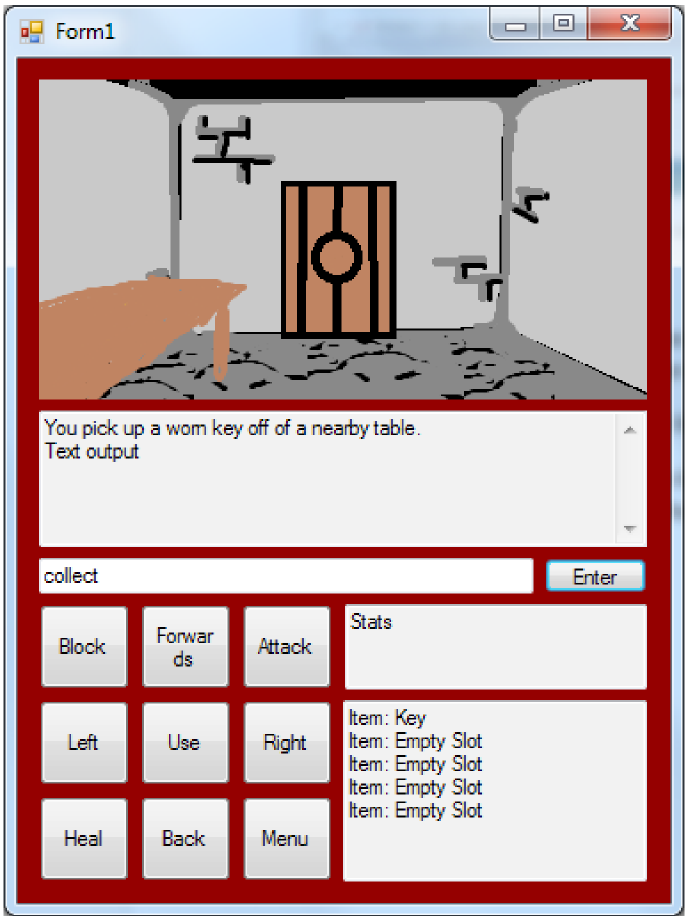
```
Error List
1 Error   0 Warnings   0 Messages

   Description                                                                                                                              File            Line   Colu...  Project
1  Invalid Resx file. Could not find file '\\CUC-SRV-FS02\Studio-StuHomeS\08houghton.e\Documents\Computing\F454\F454 Game HOME VER\WindowsFormsApplication11\Resources\Room11.png'. Line 135,   Resources.resx   135    5        WindowsFormsApplication11
   position 5.
```

Figure 49: Logic for "forward" command in room (1, 1).

I was going to run a test to see if the current code was working, however I ran into a problem. The game would not run as there was an error because the program could not find "Room11.png"

I fixed this problem by deleting the line of code in Resources.resx that referenced the file.

I then reran the test. Picking up the key in the first room worked as intended however when trying to unlock the door in room (1, 1), the game says it is an invalid action.

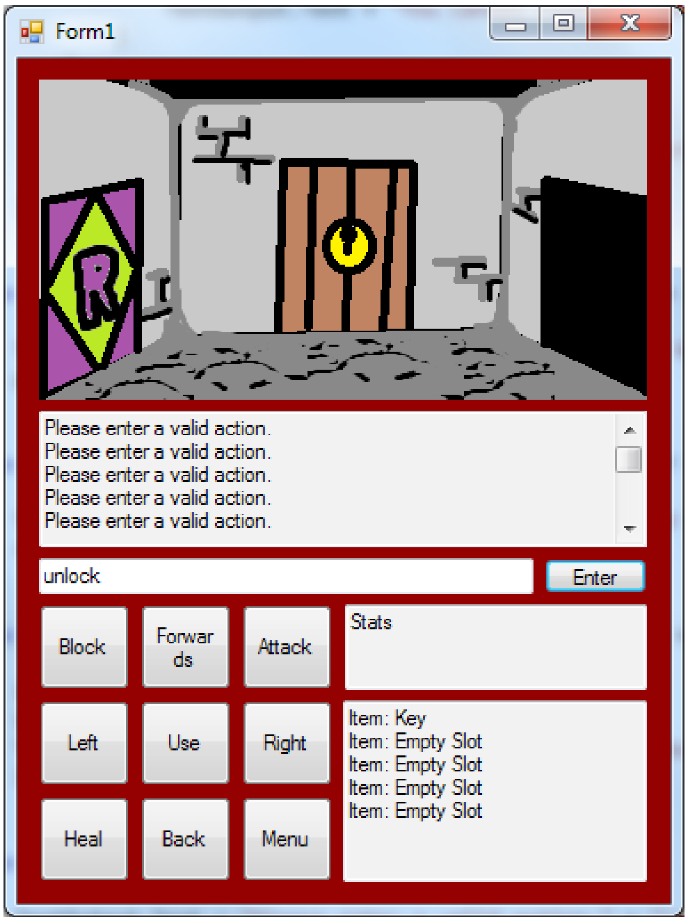


```
You pick up a worn key off of a nearby table.
Text output
```

```
collect
```

```
Item: Key
Item: Empty Slot
Item: Empty Slot
Item: Empty Slot
Item: Empty Slot
```

Figure 50: Working key "collection".



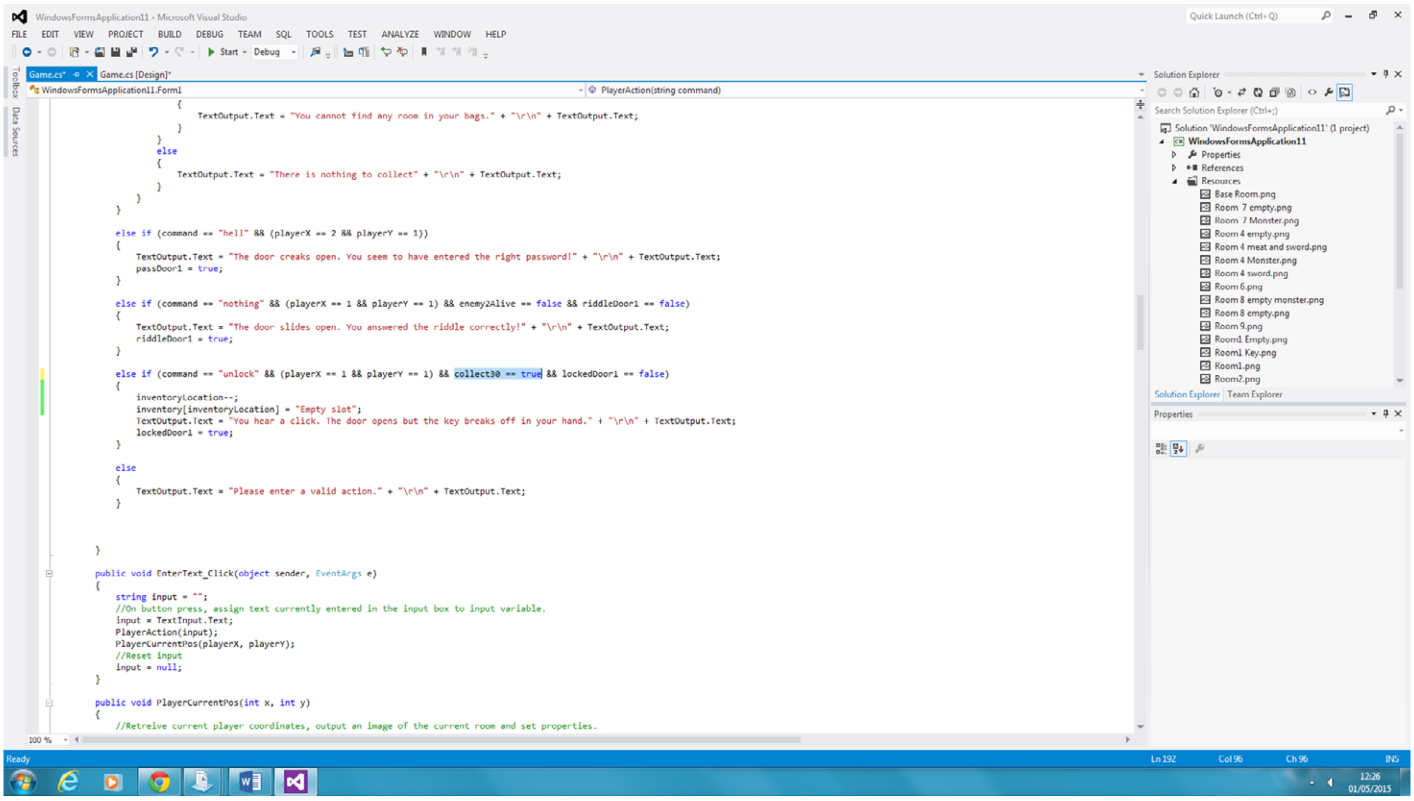
```
Please enter a valid action.
Please enter a valid action.
Please enter a valid action.
Please enter a valid action.
Please enter a valid action.
```

```
unlock
```

```
Item: Key
Item: Empty Slot
Item: Empty Slot
Item: Empty Slot
Item: Empty Slot
```

Figure 51: "unlock" command not working.

I managed to fix the problem somewhat by changing "(inventory[inventoryLocation] == "Key")" to "collect30 == true" in the "unlock" command ELSE IF statement. Now the statement will only run if the player hasn't already picked up the key.



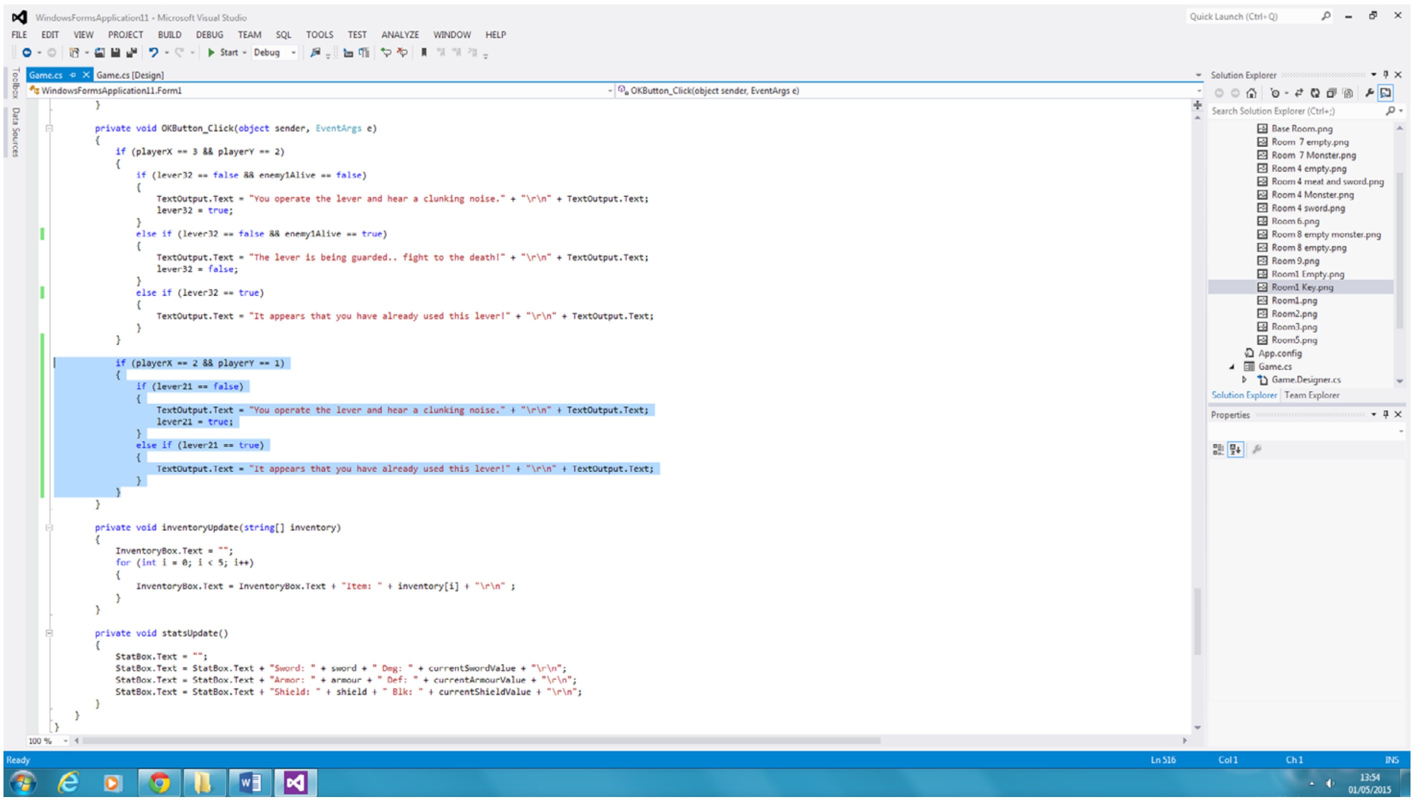
```
                        {
                            TextOutput.Text = "You cannot find any room in your bags." + "\r\n" + TextOutput.Text;
                        }
                    }
                    else
                    {
                        TextOutput.Text = "There is nothing to collect" + "\r\n" + TextOutput.Text;
                    }
                }
            }

            else if (command == "hell" && (playerX == 2 && playerY == 1))
            {
                TextOutput.Text = "The door creaks open. You seem to have entered the right password!" + "\r\n" + TextOutput.Text;
                passDoor1 = true;
            }

            else if (command == "nothing" && (playerX == 1 && playerY == 1) && enemy2Alive == false && riddleDoor1 == false)
            {
                TextOutput.Text = "The door slides open. You answered the riddle correctly!" + "\r\n" + TextOutput.Text;
                riddleDoor1 = true;
            }

            else if (command == "unlock" && (playerX == 1 && playerY == 1) && collect30 == true && lockedDoor1 == false)
            {
                inventoryLocation--;
                inventory[inventoryLocation] = "Empty slot";
                TextOutput.Text = "You hear a click. The door opens but the key breaks off in your hand." + "\r\n" + TextOutput.Text;
                lockedDoor1 = true;
            }

            else
            {
                TextOutput.Text = "Please enter a valid action." + "\r\n" + TextOutput.Text;
            }


        }

        public void EnterText_Click(object sender, EventArgs e)
        {
            string input = "";
            //On button press, assign text currently entered in the input box to input variable.
            input = TextInput.Text;
            PlayerAction(input);
            PlayerCurrentPos(playerX, playerY);
            //Reset input
            input = null;
        }

        public void PlayerCurrentPos(int x, int y)
        {
            //Retreive current player coordinates, output an image of the current room and set properties.
```

Figure 52: Changed "(inventory[inventoryLocation] == "Key")" to "collect30 == true".

Now that all the systems are complete I have to create two more enemies and change make the room (4, 1) gate require two levers to open. I will fine tune and iron out bugs afterwards.

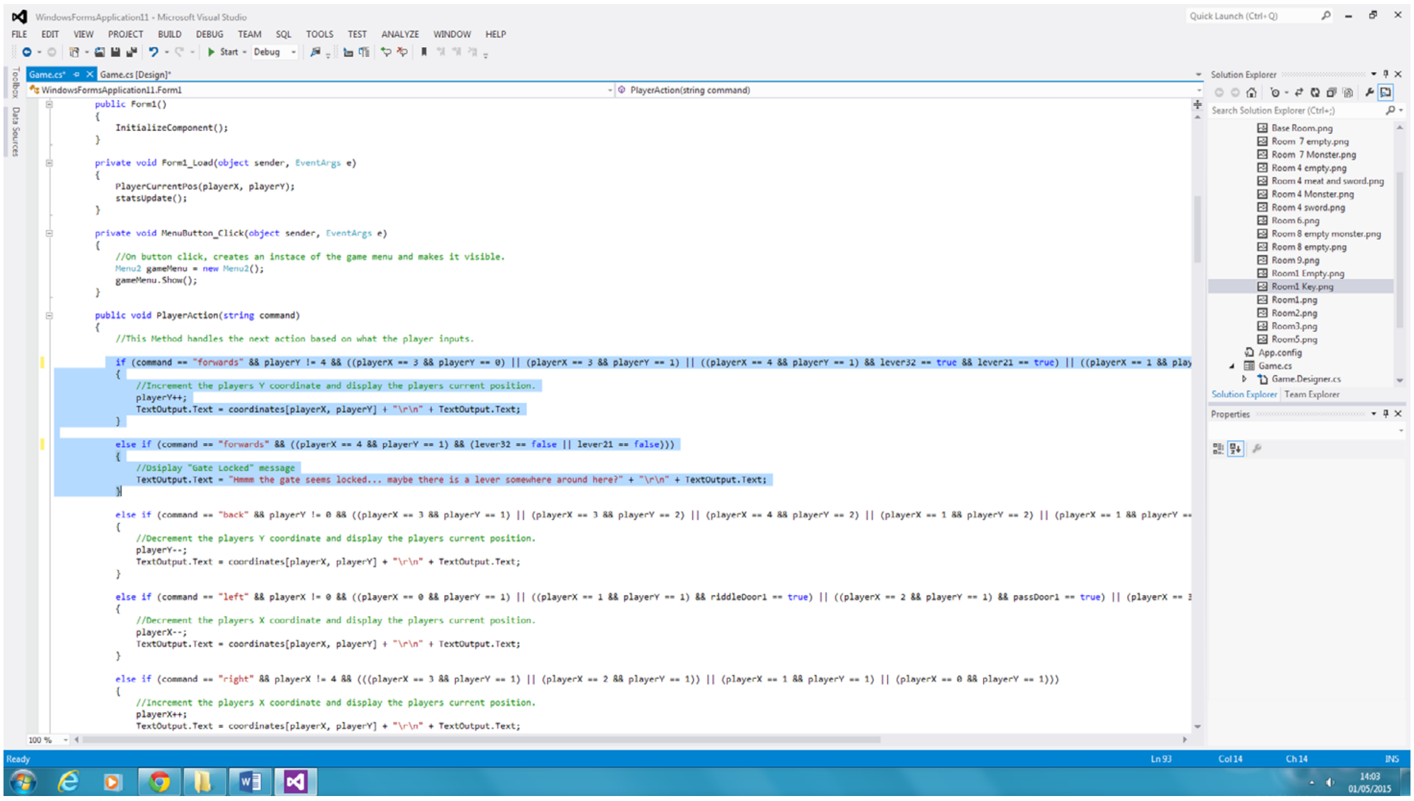
I first duplicated the code from room (3, 2)'s lever in the "OKButton_Click" method and edited it to work for the lever in room (2, 1). I then added the necessary logic to the "forward" command for room (4, 1).

I tested the gate using each lever individually and then tested both. As expected the gate only opened when both levers had been activated.



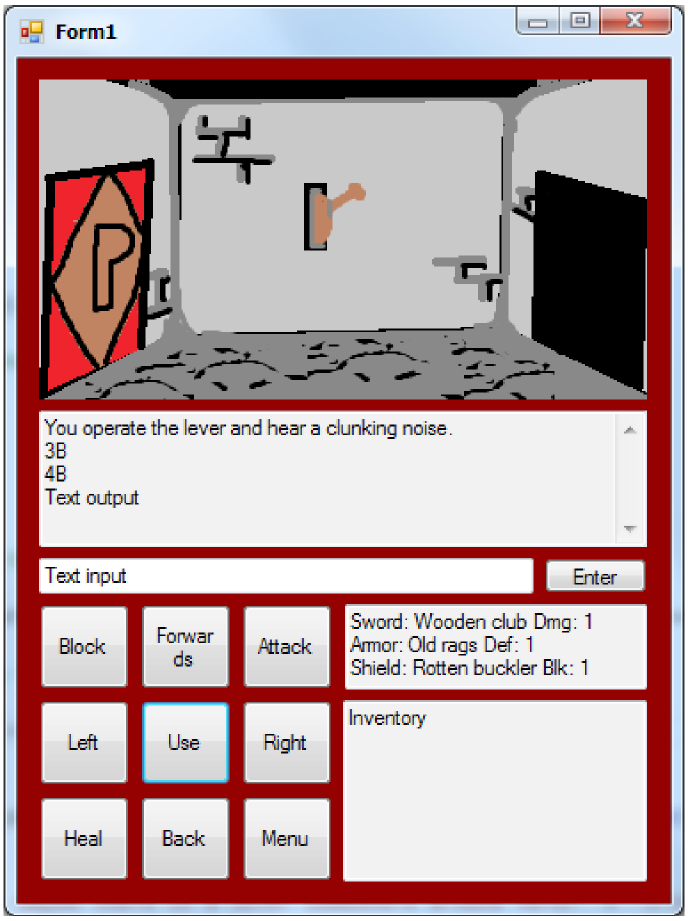
```
        }

        private void OKButton_Click(object sender, EventArgs e)
        {
            if (playerX == 3 && playerY == 2)
            {
                if (lever32 == false && enemy1Alive == false)
                {
                    TextOutput.Text = "You operate the lever and hear a clunking noise." + "\r\n" + TextOutput.Text;
                    lever32 = true;
                }
                else if (lever32 == false && enemy1Alive == true)
                {
                    TextOutput.Text = "The lever is being guarded.. fight to the death!" + "\r\n" + TextOutput.Text;
                    lever32 = false;
                }
                else if (lever32 == true)
                {
                    TextOutput.Text = "It appears that you have already used this lever!" + "\r\n" + TextOutput.Text;
                }
            }

            if (playerX == 2 && playerY == 1)
            {
                if (lever21 == false)
                {
                    TextOutput.Text = "You operate the lever and hear a clunking noise." + "\r\n" + TextOutput.Text;
                    lever21 = true;
                }
                else if (lever21 == true)
                {
                    TextOutput.Text = "It appears that you have already used this lever!" + "\r\n" + TextOutput.Text;
                }
            }
        }

        private void inventoryUpdate(string[] inventory)
        {
            InventoryBox.Text = "";
            for (int i = 0; i < 5; i++)
            {
                InventoryBox.Text = InventoryBox.Text + "Item: " + inventory[i] + "\r\n" ;
            }
        }

        private void statsUpdate()
        {
            StatBox.Text = "";
            StatBox.Text = StatBox.Text + "Sword: " + sword + " Dmg: " + currentSwordValue + "\r\n";
            StatBox.Text = StatBox.Text + "Armor: " + armour + " Def: " + currentArmourValue + "\r\n";
            StatBox.Text = StatBox.Text + "Shield: " + shield + " Blk: " + currentShieldValue + "\r\n";
        }
    }
}
```

Figure 53: Duplicating and editing code for the second lever.



```
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            PlayerCurrentPos(playerX, playerY);
            statsUpdate();
        }

        private void MenuButton_Click(object sender, EventArgs e)
        {
            //On button click, creates an instace of the game menu and makes it visible.
            Menu2 gameMenu = new Menu2();
            gameMenu.Show();
        }

        public void PlayerAction(string command)
        {
            //This Method handles the next action based on what the player inputs.

            if (command == "forwards" && playerY != 4 && ((playerX == 3 && playerY == 0) || (playerX == 3 && playerY == 1) || ((playerX == 4 && playerY == 1) && lever32 == true && lever21 == true) || ((playerX == 1 && play
            {
                //Increment the players Y coordinate and display the players current position.
                playerY++;
                TextOutput.Text = coordinates[playerX, playerY] + "\r\n" + TextOutput.Text;
            }

            else if (command == "forwards" && ((playerX == 4 && playerY == 1) && (lever32 == false || lever21 == false)))
            {
                //Dsiplay "Gate Locked" message
                TextOutput.Text = "Hmmm the gate seems locked... maybe there is a lever somewhere around here?" + "\r\n" + TextOutput.Text;
            }

            else if (command == "back" && playerY != 0 && ((playerX == 3 && playerY == 1) || (playerX == 3 && playerY == 2) || (playerX == 4 && playerY == 2) || (playerX == 1 && playerY == 2) || (playerX == 1 && playerY ==
            {
                //Decrement the players Y coordinate and display the players current position.
                playerY--;
                TextOutput.Text = coordinates[playerX, playerY] + "\r\n" + TextOutput.Text;
            }

            else if (command == "left" && playerX != 0 && ((playerX == 0 && playerY == 1) || ((playerX == 1 && playerY == 1) && riddleDoor1 == true) || ((playerX == 2 && playerY == 1) && passDoor1 == true) || (playerX == 3
            {
                //Decrement the players X coordinate and display the players current position.
                playerX--;
                TextOutput.Text = coordinates[playerX, playerY] + "\r\n" + TextOutput.Text;
            }

            else if (command == "right" && playerX != 4 && (((playerX == 3 && playerY == 1) || (playerX == 2 && playerY == 1)) || (playerX == 1 && playerY == 1) || (playerX == 0 && playerY == 1)))
            {
                //Increment the players X coordinate and display the players current position.
                playerX++;
                TextOutput.Text = coordinates[playerX, playerY] + "\r\n" + TextOutput.Text;
```

Figure 54: Adding necessary logic to "forward" command.



```
You operate the lever and hear a clunking noise.
3B
4B
Text output
```

Text input

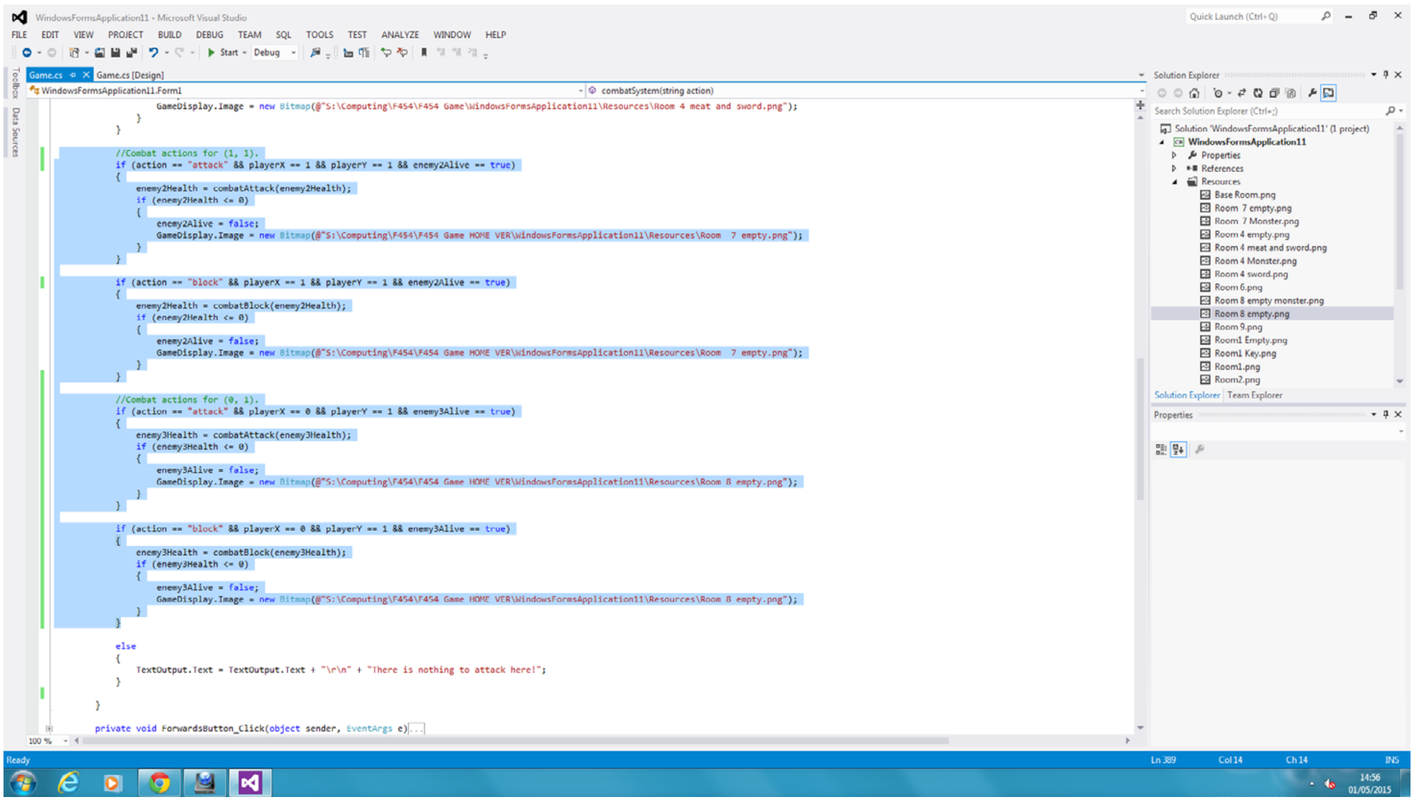
Sword: Wooden club Dmg: 1
Armor: Old rags Def: 1
Shield: Rotten buckler Blk: 1

Inventory

Figure 55: Working room (2, 1) lever.

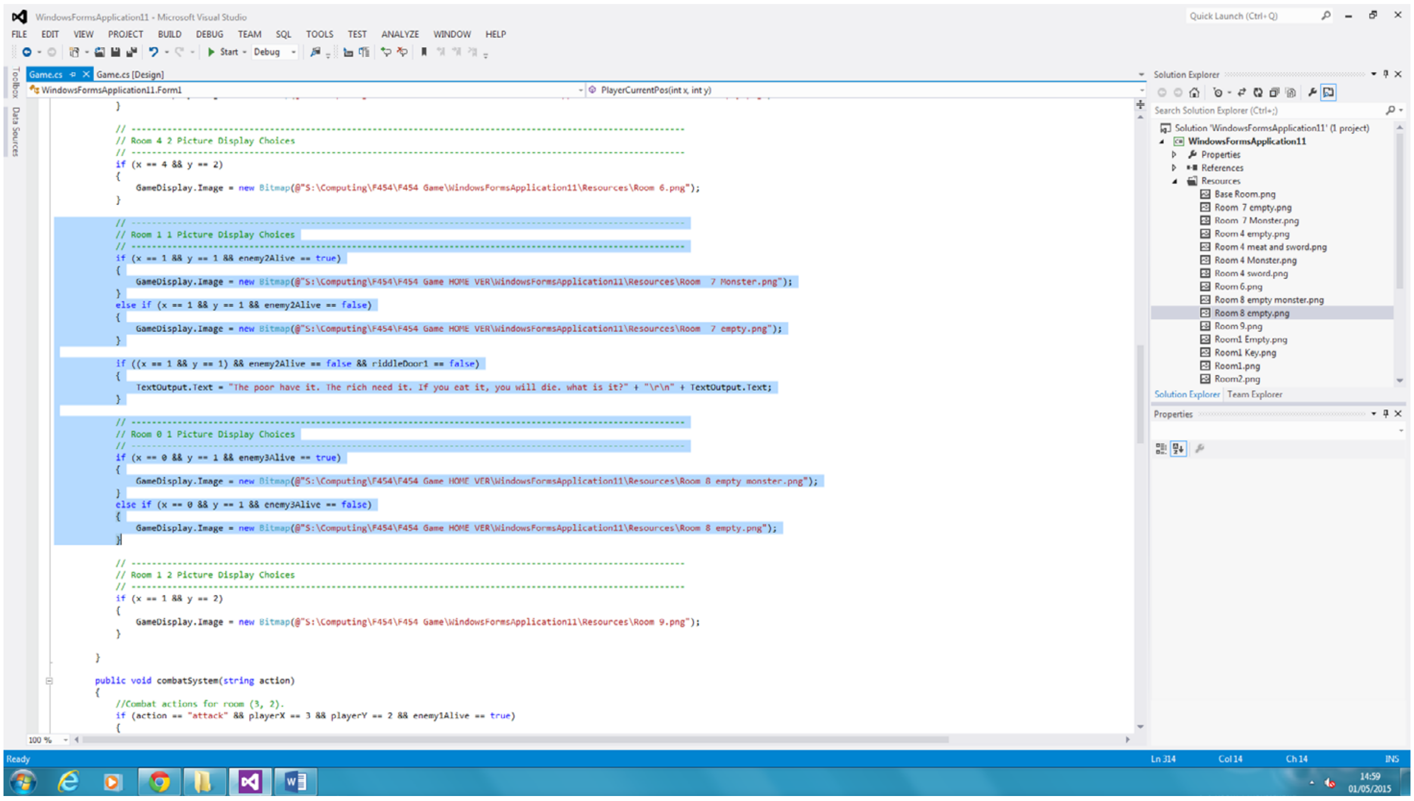
Now that this is working I need to add in the other two enemies in rooms (0, 1) and (1, 1). Similar to the levers I will duplicate the code from the first enemy and edit it to work for the other two enemies.

Once the code had been edited for the other two enemies I tested the game. The game worked as expected. I then added some code so that both enemies dropped items.



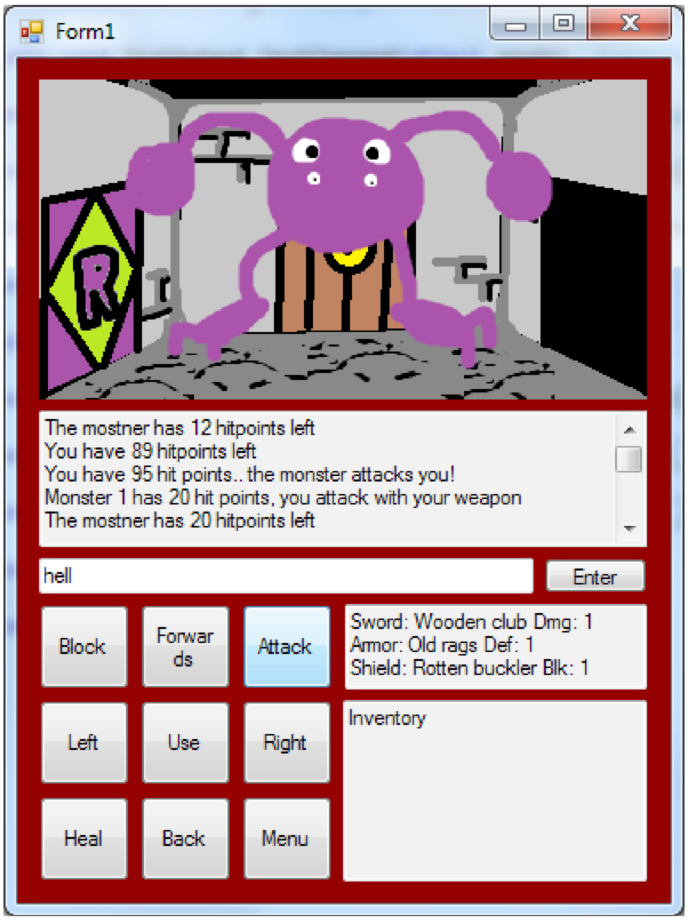
```
                GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room 4 meat and sword.png");
            }
        }

        //Combat actions for (1, 1).
        if (action == "attack" && playerX == 1 && playerY == 1 && enemy2Alive == true)
        {
            enemy2Health = combatAttack(enemy2Health);
            if (enemy2Health <= 0)
            {
                enemy2Alive = false;
                GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game HOME VER\WindowsFormsApplication11\Resources\Room  7 empty.png");
            }
        }

        if (action == "block" && playerX == 1 && playerY == 1 && enemy2Alive == true)
        {
            enemy2Health = combatBlock(enemy2Health);
            if (enemy2Health <= 0)
            {
                enemy2Alive = false;
                GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game HOME VER\WindowsFormsApplication11\Resources\Room  7 empty.png");
            }
        }

        //Combat actions for (0, 1).
        if (action == "attack" && playerX == 0 && playerY == 1 && enemy3Alive == true)
        {
            enemy3Health = combatAttack(enemy3Health);
            if (enemy3Health <= 0)
            {
                enemy3Alive = false;
                GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game HOME VER\WindowsFormsApplication11\Resources\Room 8 empty.png");
            }
        }

        if (action == "block" && playerX == 0 && playerY == 1 && enemy3Alive == true)
        {
            enemy3Health = combatBlock(enemy3Health);
            if (enemy3Health <= 0)
            {
                enemy3Alive = false;
                GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game HOME VER\WindowsFormsApplication11\Resources\Room 8 empty.png");
            }
        }

        else
        {
            TextOutput.Text = TextOutput.Text + "\r\n" + "There is nothing to attack here!";
        }

    }

    private void ForwardsButton_Click(object sender, EventArgs e)[...]
```

Figure 56: Combat action code for enemies in rooms (1, 1) and (0, 1).



```
        }

        // ------------------------------------------------------------------------------------------
        // Room 4 2 Picture Display Choices
        // ------------------------------------------------------------------------------------------
        if (x == 4 && y == 2)
        {
            GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room 6.png");
        }

        // ------------------------------------------------------------------------------------------
        // Room 1 1 Picture Display Choices
        // ------------------------------------------------------------------------------------------
        if (x == 1 && y == 1 && enemy2Alive == true)
        {
            GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game HOME VER\WindowsFormsApplication11\Resources\Room  7 Monster.png");
        }
        else if (x == 1 && y == 1 && enemy2Alive == false)
        {
            GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game HOME VER\WindowsFormsApplication11\Resources\Room  7 empty.png");
        }

        if ((x == 1 && y == 1) && enemy2Alive == false && riddleDoor1 == false)
        {
            TextOutput.Text = "The poor have it. The rich need it. If you eat it, you will die. what is it?" + "\r\n" + TextOutput.Text;
        }

        // ------------------------------------------------------------------------------------------
        // Room 0 1 Picture Display Choices
        // ------------------------------------------------------------------------------------------
        if (x == 0 && y == 1 && enemy3Alive == true)
        {
            GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game HOME VER\WindowsFormsApplication11\Resources\Room 8 empty monster.png");
        }
        else if (x == 0 && y == 1 && enemy3Alive == false)
        {
            GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game HOME VER\WindowsFormsApplication11\Resources\Room 8 empty.png");
        }

        // ------------------------------------------------------------------------------------------
        // Room 1 2 Picture Display Choices
        // ------------------------------------------------------------------------------------------
        if (x == 1 && y == 2)
        {
            GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game\WindowsFormsApplication11\Resources\Room 9.png");
        }

    }

    public void combatSystem(string action)
    {
        //Combat actions for room (3, 2).
        if (action == "attack" && playerX == 3 && playerY == 2 && enemy1Alive == true)
        {
```

Figure 57: Setting the images to be shown in the two rooms.



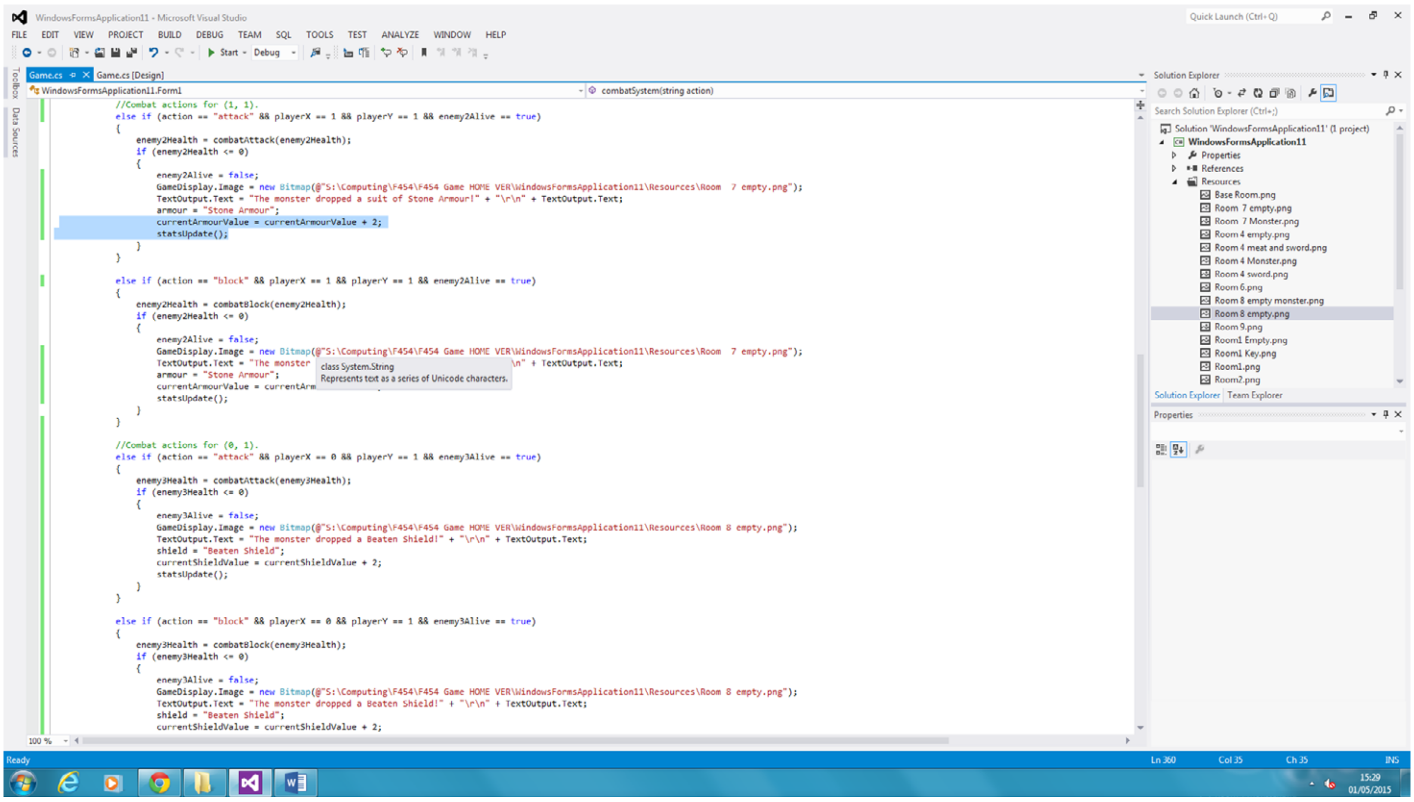
```
The mostner has 12 hitpoints left
You have 89 hitpoints left
You have 95 hit points.. the monster attacks you!
Monster 1 has 20 hit points, you attack with your weapon
The mostner has 20 hitpoints left
```

```
hell
```

```
Sword: Wooden club Dmg: 1
Armor: Old rags Def: 1
Shield: Rotten buckler Blk: 1
```

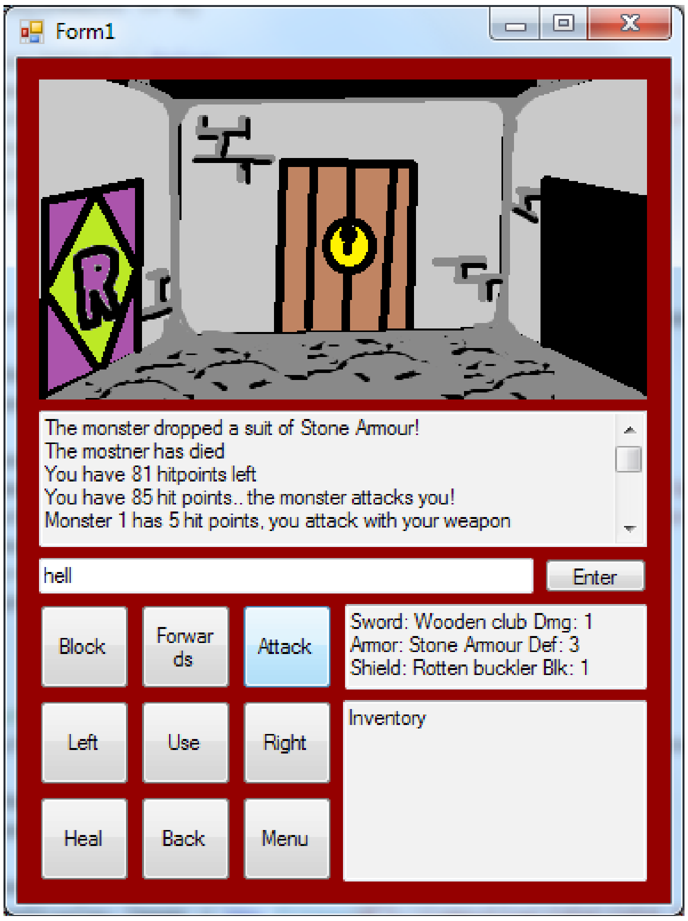
Figure 58: Working room (1, 1) enemy.

After getting the enemies working I decided to make them drop items when they are defeated. Upon being defeated the room (1, 1) enemy will drop a suit of "Stone Armour" and the room (0, 1) enemy will drop a "Beaten Shield", each one increasing their respective stats by two. After the values are changed, the "statsUpdate" method is called to reset the values in the stat box.



```
//Combat actions for (1, 1).
else if (action == "attack" && playerX == 1 && playerY == 1 && enemy2Alive == true)
{
    enemy2Health = combatAttack(enemy2Health);
    if (enemy2Health <= 0)
    {
        enemy2Alive = false;
        GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game HOME VER\WindowsFormsApplication11\Resources\Room  7 empty.png");
        TextOutput.Text = "The monster dropped a suit of Stone Armour!" + "\r\n" + TextOutput.Text;
        armour = "Stone Armour";
        currentArmourValue = currentArmourValue + 2;
        statsUpdate();
    }
}

else if (action == "block" && playerX == 1 && playerY == 1 && enemy2Alive == true)
{
    enemy2Health = combatBlock(enemy2Health);
    if (enemy2Health <= 0)
    {
        enemy2Alive = false;
        GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game HOME VER\WindowsFormsApplication11\Resources\Room  7 empty.png");
        TextOutput.Text = "The monster [class System.String Represents text as a series of Unicode characters.] \n" + TextOutput.Text;
        armour = "Stone Armour";
        currentArmourValue = currentArm
        statsUpdate();
    }
}

//Combat actions for (0, 1).
else if (action == "attack" && playerX == 0 && playerY == 1 && enemy3Alive == true)
{
    enemy3Health = combatAttack(enemy3Health);
    if (enemy3Health <= 0)
    {
        enemy3Alive = false;
        GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game HOME VER\WindowsFormsApplication11\Resources\Room 8 empty.png");
        TextOutput.Text = "The monster dropped a Beaten Shield!" + "\r\n" + TextOutput.Text;
        shield = "Beaten Shield";
        currentShieldValue = currentShieldValue + 2;
        statsUpdate();
    }
}

else if (action == "block" && playerX == 0 && playerY == 1 && enemy3Alive == true)
{
    enemy3Health = combatBlock(enemy3Health);
    if (enemy3Health <= 0)
    {
        enemy3Alive = false;
        GameDisplay.Image = new Bitmap(@"S:\Computing\F454\F454 Game HOME VER\WindowsFormsApplication11\Resources\Room 8 empty.png");
        TextOutput.Text = "The monster dropped a Beaten Shield!" + "\r\n" + TextOutput.Text;
        shield = "Beaten Shield";
        currentShieldValue = currentShieldValue + 2;
```

Figure 59: Adding code for item drops.



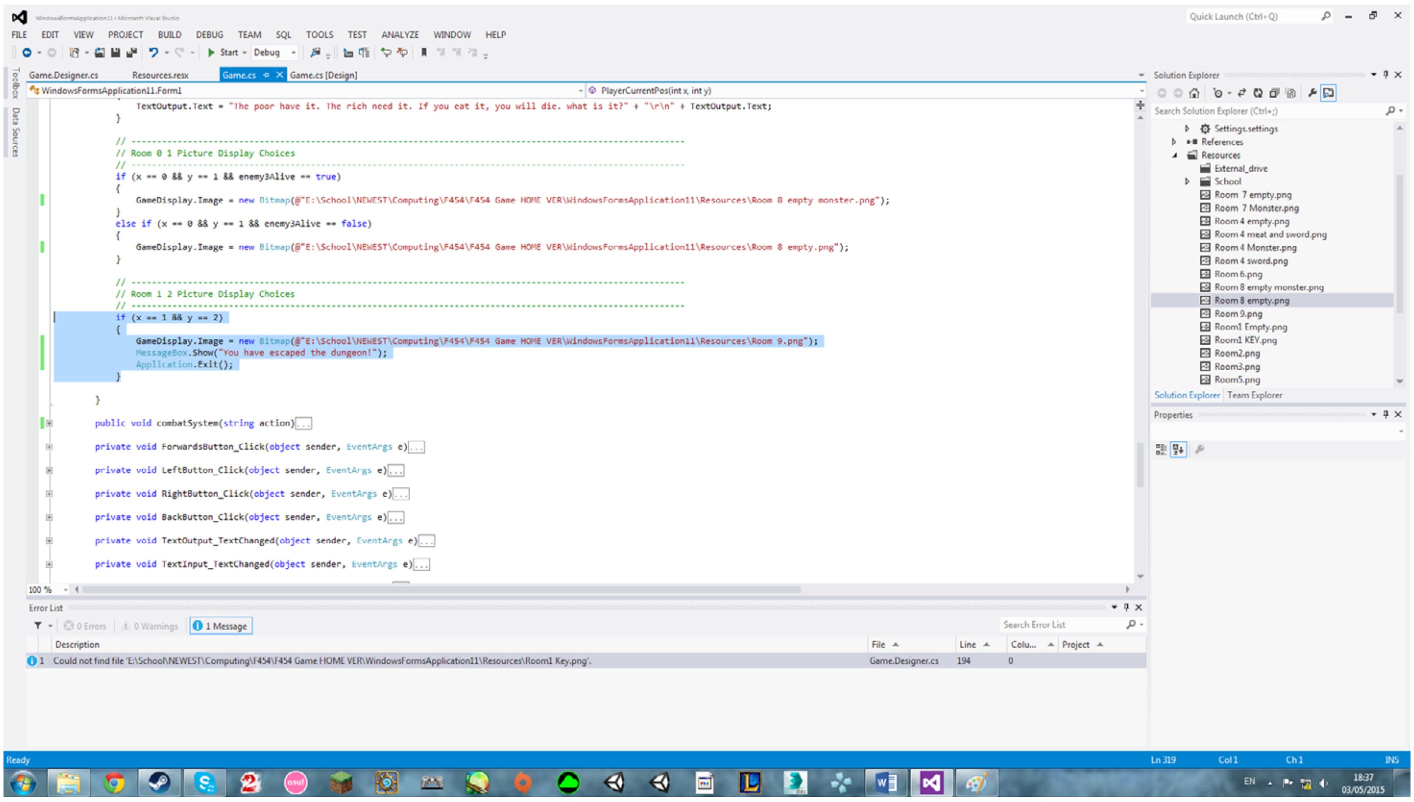
```
The monster dropped a suit of Stone Armour!
The mostner has died
You have 81 hitpoints left
You have 85 hit points.. the monster attacks you!
Monster 1 has 5 hit points, you attack with your weapon
```

```
hell
```

```
Sword: Wooden club Dmg: 1
Armor: Stone Armour Def: 3
Shield: Rotten buckler Blk: 1
```

Figure 60: Working item drops.

Now that the game is effectively finished, I need to add some sort of end state. I will do this by opening a popup window telling the player that they “escaped the dungeon!” or something along those lines once they reach room (1, 2).

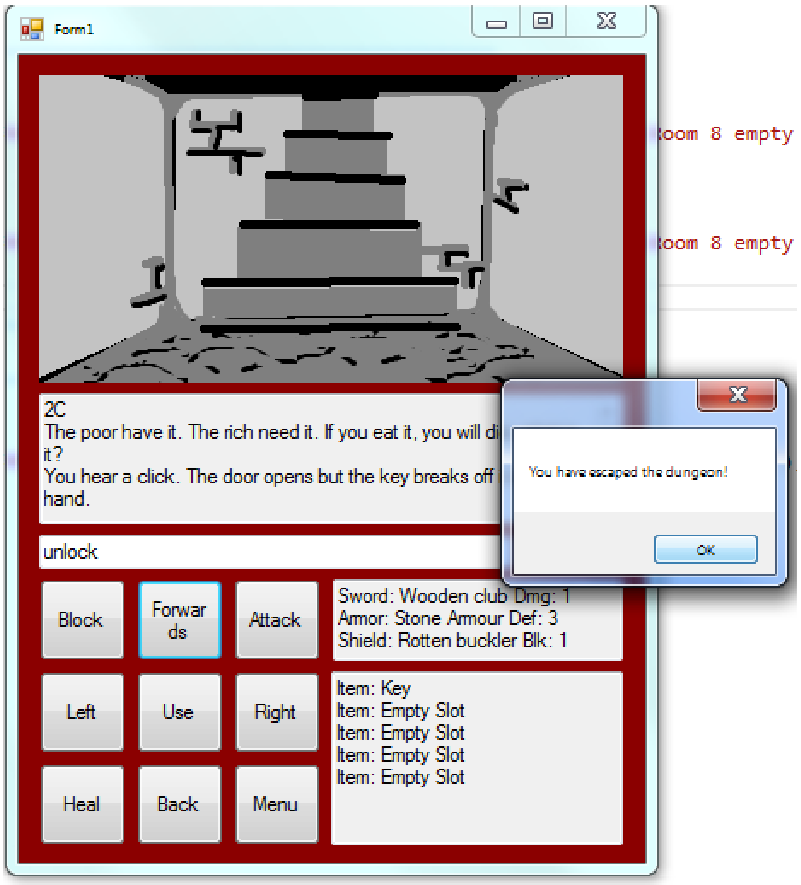


```
                TextOutput.Text = "The poor have it. The rich need it. If you eat it, you will die. what is it?" + "\r\n" + TextOutput.Text;
            }

            // -----------------------------------------------------------------------------------------
            // Room 0 1 Picture Display Choices
            // -----------------------------------------------------------------------------------------
            if (x == 0 && y == 1 && enemy3Alive == true)
            {
                GameDisplay.Image = new Bitmap(@"E:\School\NEWEST\Computing\F454\F454 Game HOME VER\WindowsFormsApplication11\Resources\Room 8 empty monster.png");
            }
            else if (x == 0 && y == 1 && enemy3Alive == false)
            {
                GameDisplay.Image = new Bitmap(@"E:\School\NEWEST\Computing\F454\F454 Game HOME VER\WindowsFormsApplication11\Resources\Room 8 empty.png");
            }

            // -----------------------------------------------------------------------------------------
            // Room 1 2 Picture Display Choices
            // -----------------------------------------------------------------------------------------
            if (x == 1 && y == 2)
            {
                GameDisplay.Image = new Bitmap(@"E:\School\NEWEST\Computing\F454\F454 Game HOME VER\WindowsFormsApplication11\Resources\Room 9.png");
                MessageBox.Show("You have escaped the dungeon!");
                Application.Exit();
            }

        }

        public void combatSystem(string action)[...]

        private void ForwardsButton_Click(object sender, EventArgs e)[...]

        private void LeftButton_Click(object sender, EventArgs e)[...]

        private void RightButton_Click(object sender, EventArgs e)[...]

        private void BackButton_Click(object sender, EventArgs e)[...]

        private void TextOutput_TextChanged(object sender, EventArgs e)[...]

        private void TextInput_TextChanged(object sender, EventArgs e)[...]
```

```
Error List
0 Errors   0 Warnings   1 Message
   Description                                                                                                         File               Line   Column  Project
1  Could not find file 'E:\School\NEWEST\Computing\F454\F454 Game HOME VER\WindowsFormsApplication11\Resources\Room1 Key.png'.   Game.Designer.cs   194    0
```

Figure 61: End state code.

Upon entering the final room, a pop up box appears telling the player "You have escaped the dungeon!". Once the player clicks ok the application closes.



```
2C
The poor have it. The rich need it. If you eat it, you will di
it?
You hear a click. The door opens but the key breaks off
hand.
```

```
unlock
```

```
Sword: Wooden club Dmg: 1
Armor: Stone Armour Def: 3
Shield: Rotten buckler Blk: 1
```

```
Item: Key
Item: Empty Slot
Item: Empty Slot
Item: Empty Slot
Item: Empty Slot
```

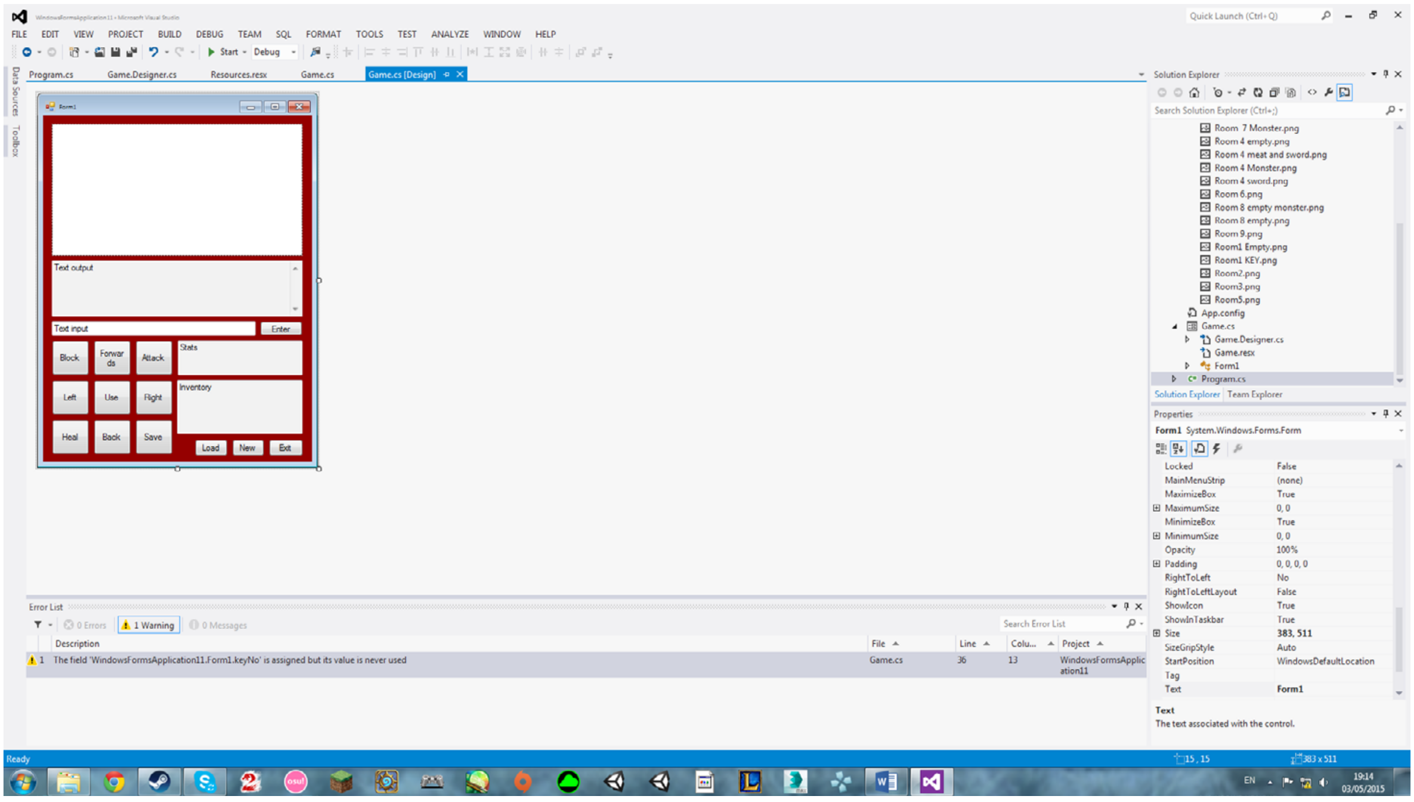
```
You have escaped the dungeon!
```

Figure 62: Working end state message.

The final thing I need to do before my application is complete is add a save/load function. This will work by writing the current values of all global variables to a text file whenever the player clicks the save button. When the player clicks the load button from the menu, the game will start and set all the variables to the values written in the text file.

I realised that even though I have three other forms besides the game screen that the player must navigate, there isn't actually anything on those screens that could not be relocated to the game screen itself so I decided to scrap the other forms and just have "New", "Save" and "Load" buttons below the inventory box. This would also mean scrapping the options menu which at this point was rather unrealistic anyway, and also changing the menu button to another function.

The game screen I ended up with after relocating and repurposing buttons is shown in the image below. Having all necessary functions available on one screen saves the user from wasting time navigating menus. The options on the options menu were unnecessary and did not require a screen to themselves so I scrapped the save location and screen size options along with the options form.



Error List

0 Errors | 1 Warning | 0 Messages

	Description	File	Line	Colu...	Project
1	The field 'WindowsFormsApplication11.Form1.keyNo' is assigned but its value is never used	Game.cs	36	13	WindowsFormsApplication11

Properties

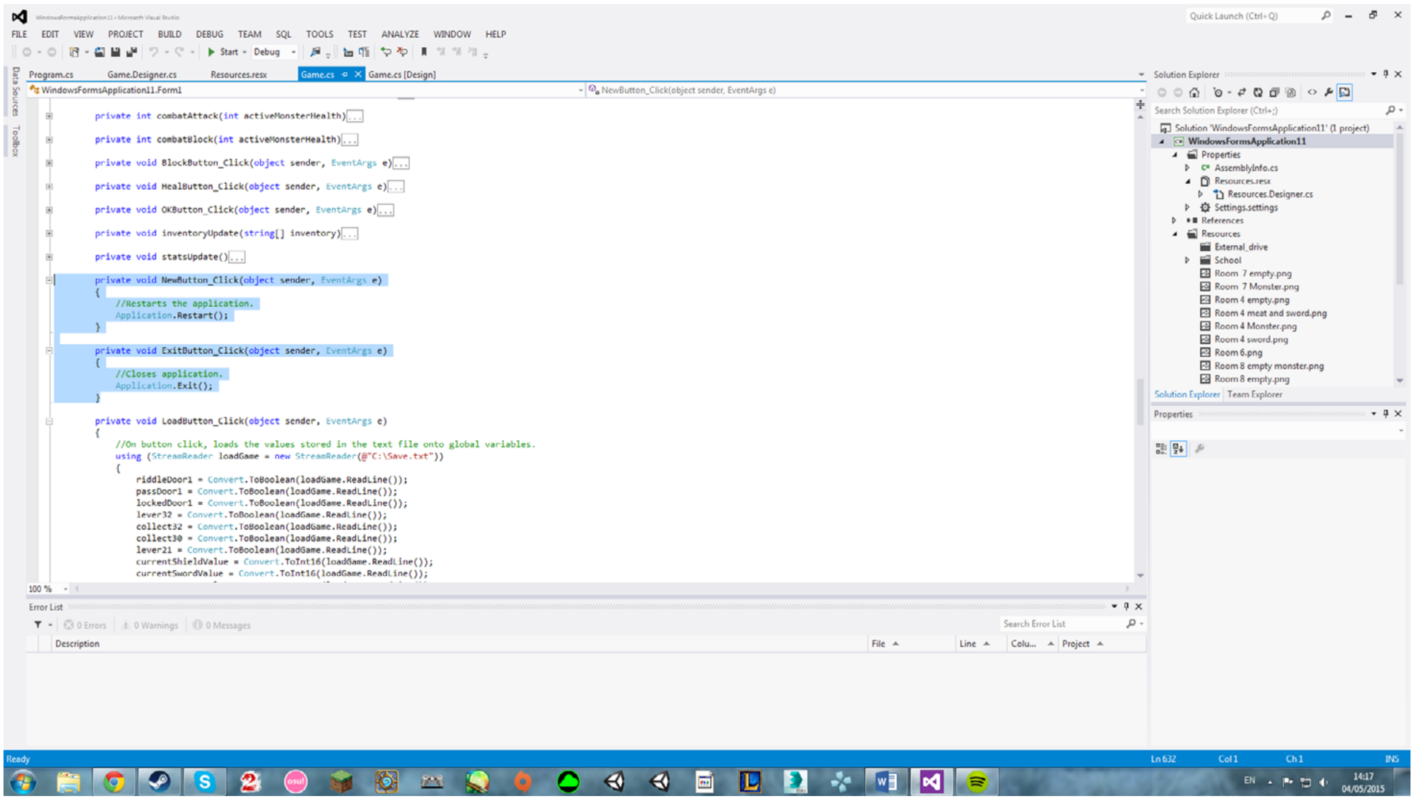
Form1 System.Windows.Forms.Form

Locked	False
MainMenuStrip	(none)
MaximizeBox	True
MaximumSize	0, 0
MinimizeBox	True
MinimumSize	0, 0
Opacity	100%
Padding	0, 0, 0, 0
RightToLeft	No
RightToLeftLayout	False
ShowIcon	True
ShowInTaskbar	True
Size	**383, 511**
SizeGripStyle	Auto
StartPosition	WindowsDefaultLocation
Tag	
Text	**Form1**

Text
The text associated with the control.

Figure 63: Final version of the game screen layout.

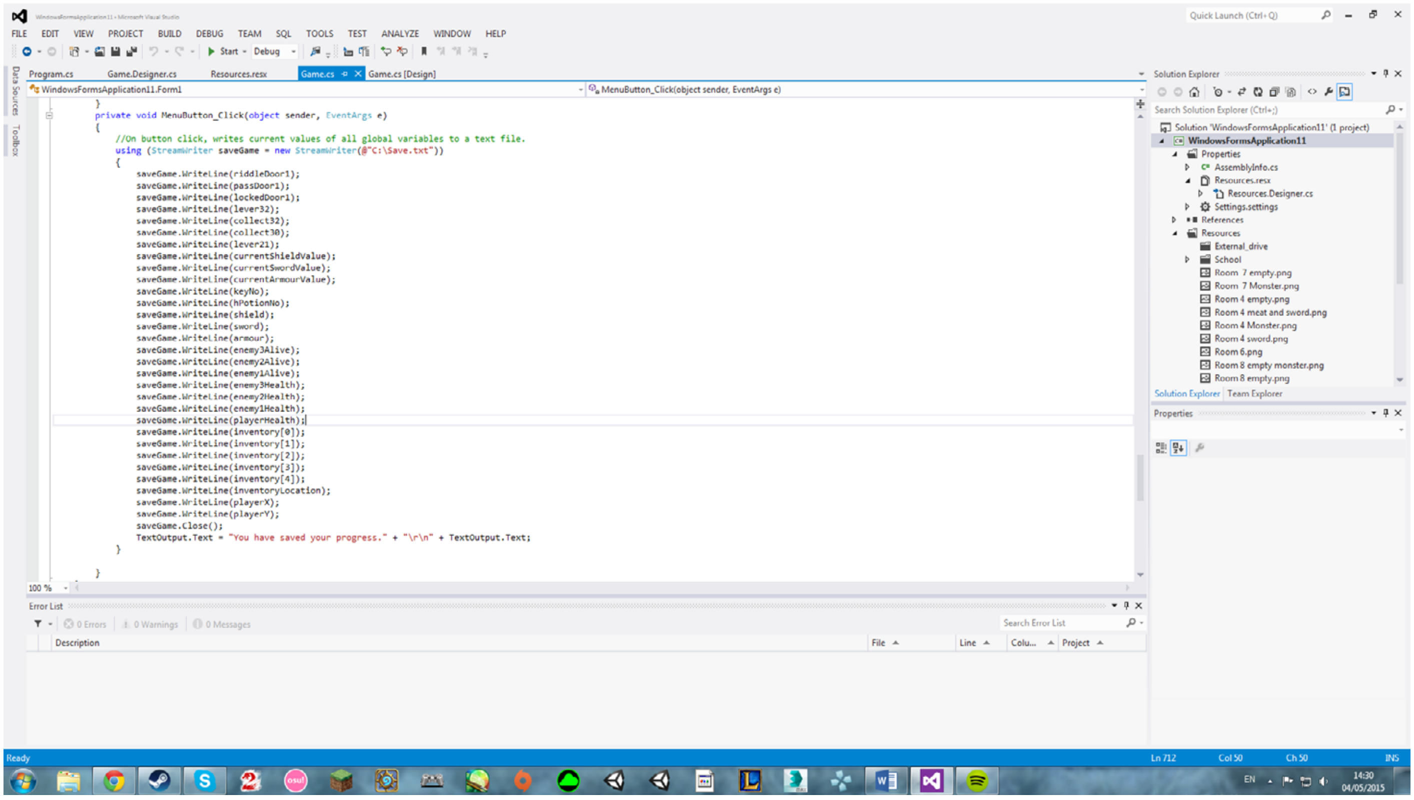
I added in the code for the "Exit" and "New" buttons first. The exit button simply closes the application and the "New" button restarts the application with its default values.



```
        private int combatAttack(int activeMonsterHealth)...

        private int combatBlock(int activeMonsterHealth)...

        private void BlockButton_Click(object sender, EventArgs e)...

        private void HealButton_Click(object sender, EventArgs e)...

        private void OKButton_Click(object sender, EventArgs e)...

        private void inventoryUpdate(string[] inventory)...

        private void statsUpdate()...

        private void NewButton_Click(object sender, EventArgs e)
        {
            //Restarts the application.
            Application.Restart();
        }

        private void ExitButton_Click(object sender, EventArgs e)
        {
            //Closes application.
            Application.Exit();
        }

        private void LoadButton_Click(object sender, EventArgs e)
        {
            //On button click, loads the values stored in the text file onto global variables.
            using (StreamReader loadGame = new StreamReader(@"C:\Save.txt"))
            {
                riddleDoor1 = Convert.ToBoolean(loadGame.ReadLine());
                passDoor1 = Convert.ToBoolean(loadGame.ReadLine());
                lockedDoor1 = Convert.ToBoolean(loadGame.ReadLine());
                lever32 = Convert.ToBoolean(loadGame.ReadLine());
                collect32 = Convert.ToBoolean(loadGame.ReadLine());
                collect30 = Convert.ToBoolean(loadGame.ReadLine());
                lever21 = Convert.ToBoolean(loadGame.ReadLine());
                currentShieldValue = Convert.ToInt16(loadGame.ReadLine());
                currentSwordValue = Convert.ToInt16(loadGame.ReadLine());
```

Figure 64: "Exit" and "New" button code.

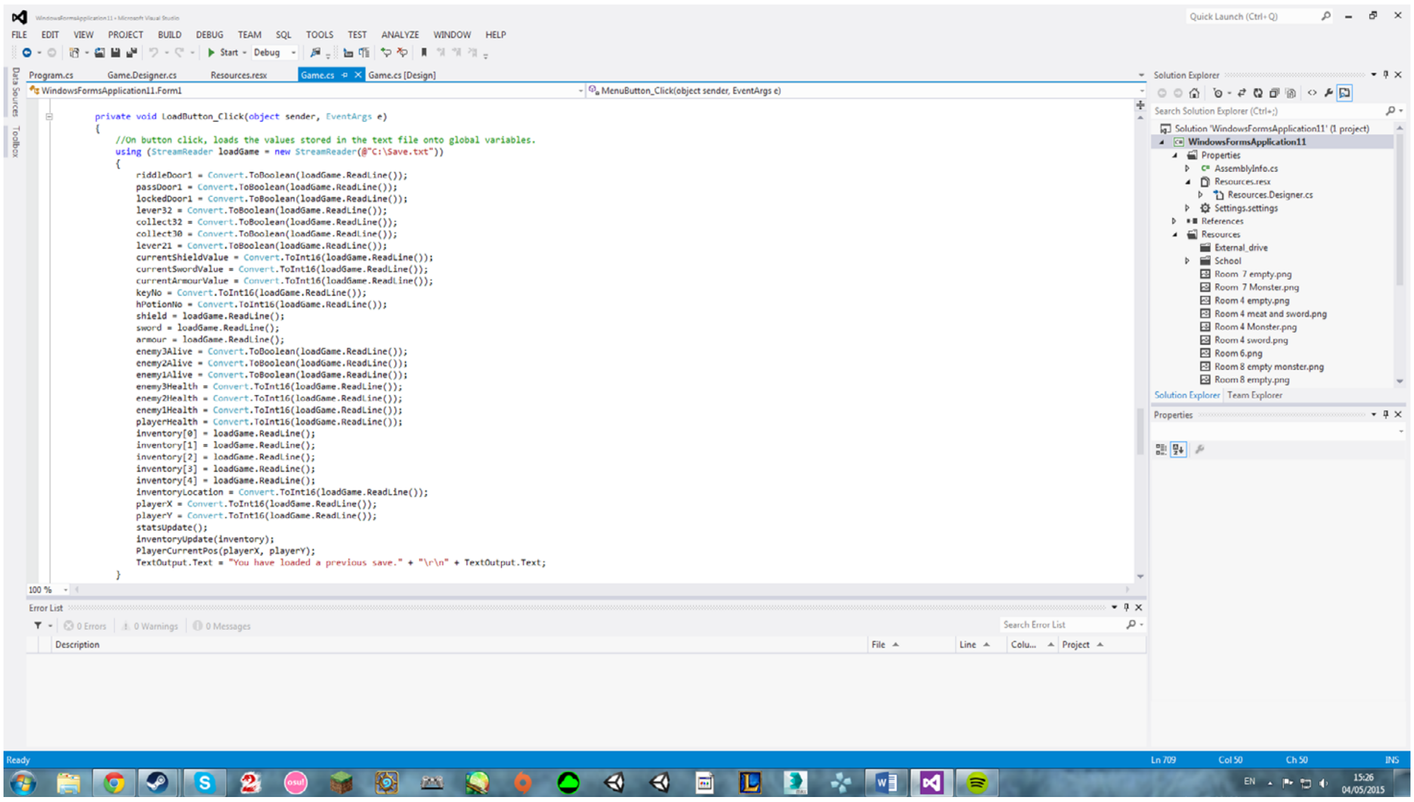
After that I created the "Save" button. When the user clicks the button all the global variables are written to a text file called "Save.txt" located in the C: drive. The file is then closed and the user is informed that they have saved the game.



```
        }
        private void MenuButton_Click(object sender, EventArgs e)
        {
            //On button click, writes current values of all global variables to a text file.
            using (StreamWriter saveGame = new StreamWriter(@"C:\Save.txt"))
            {
                saveGame.WriteLine(riddleDoor1);
                saveGame.WriteLine(passDoor1);
                saveGame.WriteLine(lockedDoor1);
                saveGame.WriteLine(lever32);
                saveGame.WriteLine(collect32);
                saveGame.WriteLine(collect30);
                saveGame.WriteLine(lever21);
                saveGame.WriteLine(currentShieldValue);
                saveGame.WriteLine(currentSwordValue);
                saveGame.WriteLine(currentArmourValue);
                saveGame.WriteLine(keyNo);
                saveGame.WriteLine(hPotionNo);
                saveGame.WriteLine(shield);
                saveGame.WriteLine(sword);
                saveGame.WriteLine(armour);
                saveGame.WriteLine(enemy3Alive);
                saveGame.WriteLine(enemy2Alive);
                saveGame.WriteLine(enemy1Alive);
                saveGame.WriteLine(enemy3Health);
                saveGame.WriteLine(enemy2Health);
                saveGame.WriteLine(enemy1Health);
                saveGame.WriteLine(playerHealth);
                saveGame.WriteLine(inventory[0]);
                saveGame.WriteLine(inventory[1]);
                saveGame.WriteLine(inventory[2]);
                saveGame.WriteLine(inventory[3]);
                saveGame.WriteLine(inventory[4]);
                saveGame.WriteLine(inventoryLocation);
                saveGame.WriteLine(playerX);
                saveGame.WriteLine(playerY);
                saveGame.Close();
                TextOutput.Text = "You have saved your progress." + "\r\n" + TextOutput.Text;
            }

        }
```

Figure 65: "Save" button code.

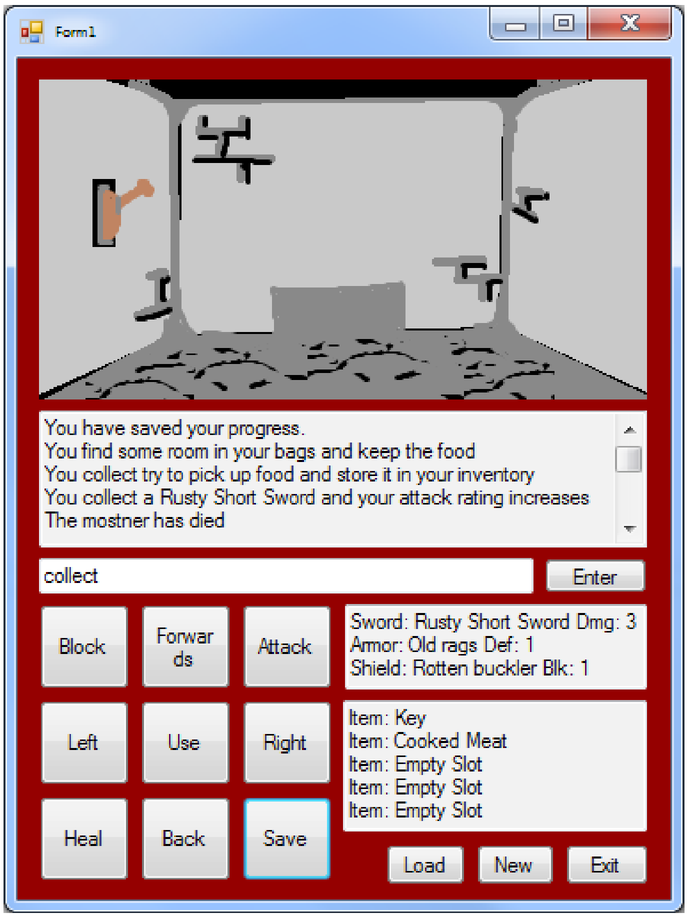
Lastly I created the "Load" button code. When the user clicks the button every global variable is assigned the contents of the "Save.txt" file. As the contents of the file are stored as strings, they are converted into the necessary data types before being assigned to the variables. The "inventoryUpdate", "statsUpdate" and "PlayerCurrentPos" methods are then called so all the outputs are showing the correct values. The player is then informed that they have loaded a previous save.



```
        private void LoadButton_Click(object sender, EventArgs e)
        {
            //On button click, loads the values stored in the text file onto global variables.
            using (StreamReader loadGame = new StreamReader(@"C:\Save.txt"))
            {
                riddleDoor1 = Convert.ToBoolean(loadGame.ReadLine());
                passDoor1 = Convert.ToBoolean(loadGame.ReadLine());
                lockedDoor1 = Convert.ToBoolean(loadGame.ReadLine());
                lever32 = Convert.ToBoolean(loadGame.ReadLine());
                collect32 = Convert.ToBoolean(loadGame.ReadLine());
                collect30 = Convert.ToBoolean(loadGame.ReadLine());
                lever21 = Convert.ToBoolean(loadGame.ReadLine());
                currentShieldValue = Convert.ToInt16(loadGame.ReadLine());
                currentSwordValue = Convert.ToInt16(loadGame.ReadLine());
                currentArmourValue = Convert.ToInt16(loadGame.ReadLine());
                keyNo = Convert.ToInt16(loadGame.ReadLine());
                hPotionNo = Convert.ToInt16(loadGame.ReadLine());
                shield = loadGame.ReadLine();
                sword = loadGame.ReadLine();
                armour = loadGame.ReadLine();
                enemy3Alive = Convert.ToBoolean(loadGame.ReadLine());
                enemy2Alive = Convert.ToBoolean(loadGame.ReadLine());
                enemy1Alive = Convert.ToBoolean(loadGame.ReadLine());
                enemy3Health = Convert.ToInt16(loadGame.ReadLine());
                enemy2Health = Convert.ToInt16(loadGame.ReadLine());
                enemy1Health = Convert.ToInt16(loadGame.ReadLine());
                playerHealth = Convert.ToInt16(loadGame.ReadLine());
                inventory[0] = loadGame.ReadLine();
                inventory[1] = loadGame.ReadLine();
                inventory[2] = loadGame.ReadLine();
                inventory[3] = loadGame.ReadLine();
                inventory[4] = loadGame.ReadLine();
                inventoryLocation = Convert.ToInt16(loadGame.ReadLine());
                playerX = Convert.ToInt16(loadGame.ReadLine());
                playerY = Convert.ToInt16(loadGame.ReadLine());
                statsUpdate();
                inventoryUpdate(inventory);
                PlayerCurrentPos(playerX, playerY);
                TextOutput.Text = "You have loaded a previous save." + "\r\n" + TextOutput.Text;
            }
```

Figure 66: "Load" button code.

After running a test, the buttons all worked as intended. I ran through a portion of the game, saved and exited. I then reran the game and clicked load which immediately put me back where I was.



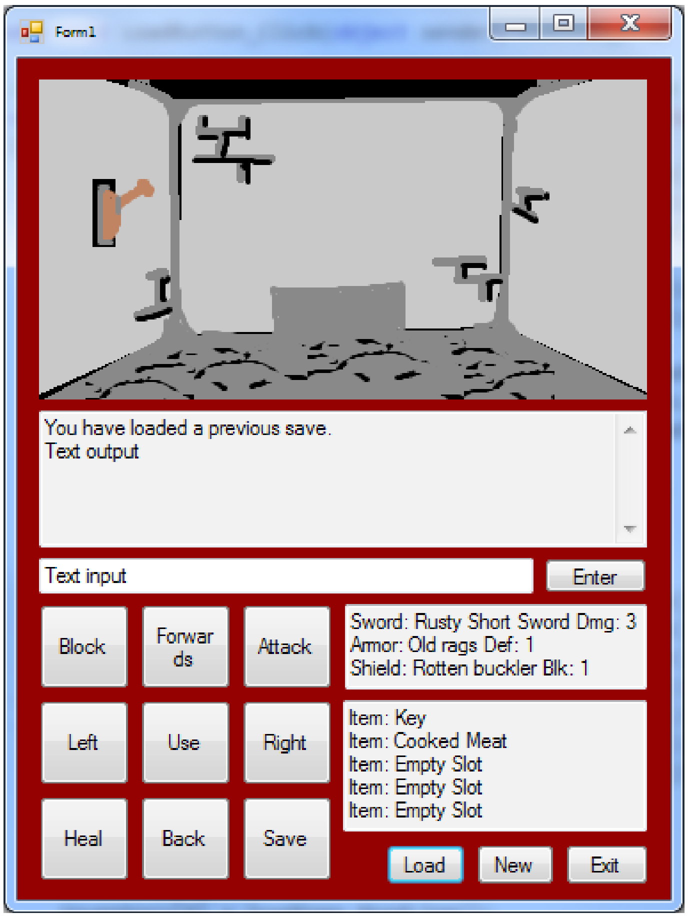
```
You have saved your progress.
You find some room in your bags and keep the food
You collect try to pick up food and store it in your inventory
You collect a Rusty Short Sword and your attack rating increases
The mostner has died
```

collect

Sword: Rusty Short Sword Dmg: 3
Armor: Old rags Def: 1
Shield: Rotten buckler Blk: 1

Item: Key
Item: Cooked Meat
Item: Empty Slot
Item: Empty Slot
Item: Empty Slot

Figure 67: Working "save" button.



You have loaded a previous save.
Text output

Text input

Sword: Rusty Short Sword Dmg: 3
Armor: Old rags Def: 1
Shield: Rotten buckler Blk: 1

Item: Key
Item: Cooked Meat
Item: Empty Slot
Item: Empty Slot
Item: Empty Slot

Figure 68: Working "Load" button.

The program is now finished. Now I will move on to testing.

Testing

The program must be now be tested by myself and the end user to see if the game functions as it should and identify any major bugs. I will be presenting my findings in test table format to increase efficiency and readability. The test will cover movement and input/output.

Shown below are the results of my test. The table is rather large as I wanted to be thorough with my testing.

My test

Test Case	ScenariExpected results	Actual results	Pass/Fail	Evidence (Optional)
1)	**Movement** – Testing that the restrictions on movement are working correctly.			
1.1)	Room1-Right	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	Room1-Left	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	Room1-Back	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	Room1-Forward	Move to room 2	Move to room 2	Pass
1.2)	Room2-Right	Move to room 5	Move to room 5	Pass

Test Case	ScenariExpected results	Actual results	Pass/Fail	Evidence (Optional)
	Room2-Left	Move to room 3	Move to room 3	Pass
	Room2-Back	Move to room 1	Move to room 1	Pass
	Room2-Forward	Move to room 4	Move to room 4	Pass
1.3)	Room3-Right	Move to room 2	Move to room 2	Pass
	Room3-Left	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	Room3-Left and password entered correctly	Move to room 7	Move to room 7	Pass
	Room3-Back	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	Room3-Forward	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
1.4)	Room4-Right	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	Room4-Left	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	Room4-Back	Move to room 2	Move to room 2	Pass
	Room4-Forward	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
1.5)	Room5-Right	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	Room5-Left	Move to room 2	Move to room 2	Pass
	Room5-Back	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	Room5-Forward	Output "Hmmm the gate seems locked... maybe there is a lever somewhere around here?"	Output "Hmmm the gate seems locked... maybe there is a lever somewhere around here?"	Pas
	Room5-Forward and both levers are activated	Move to room 6	Move to room 6	Pass
1.6)	Room6-Right	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	Room6-Left	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	Room6-Back	Move to room 5	Move to room 5	Pass
	Room6-Forward	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
1.7)	Room7-Right	Move to room 3	Move to room 3	Pass
	Room7-Left	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	Room7-Left and riddle answered correctly	Move to room 8	Move to room 8	Pass
	Room7-Back	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass

Test Case	ScenariExpected results	Actual results	Pass/Fail	Evidence (Optional)
	Room7-Forward	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	Room7-Forward and locked door unlocked	Move to room 9	Move to room 9	Pass
	Room7-Enter	Output "The poor have it. The rich need it. If you eat it, you will die. what is it?"	Output "The poor have it. The rich need it. If you eat it, you will die. what is it?"	Pass
1.8)	Room8-Right	Move to room 7	Move to room 7	Pass
	Room8-Left	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	Room8-Back	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	Room8-Forward	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
2)	**Input/output –** Testing that all inputs provide the correct results.			
2.1)	"collect" (Lower-case) command in room (3, 0)	Add key to inventory, change room (3, 0) image	Add key to inventory, change room (3, 0) image	Pass
	"collect" (Lower-case) command in room (3, 2) with enemy alive	Output "There is nothing to collect here... try killing the monster!!"	Output "There is nothing to collect here... try killing the monster!!"	Pass
	"collect" (Lower-case) command in room (3, 2) with enemy dead	Add cooked meat to inventory, add 2 to sword dmg	Add cooked meat to inventory, add 2 to sword dmg	Pass
	"collect" (Lower-case) command outside of rooms (3, 0) and (3, 2)	Output "Please enter a valid action"	Nothing	Fail
	"Collect" (Upper-case) command in room (3, 0)	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	"Collect" (Upper-case) command in room (3, 2) with enemy alive	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	"Collect" (Upper-case) command in room (3, 2) with enemy dead	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	"Collect" (Upper-case) command outside of rooms (3, 0) and (3, 2)	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
2.2)	"Use" button in room (3, 2) with enemy alive	Output "The lever is being guarded.. fight to the death!"	Output "The lever is being guarded.. fight to the death!"	Pass
	"Use" button in room (3, 2) with enemy dead	Activate lever 1, output "You operate the lever and hear a clunking noise."	Activate lever 1, output "You operate the lever and hear a clunking noise."	Pass

Test Case	ScenariExpected results	Actual results	Pass/Fail	Evidence (Optional)
	"Use" button in room (2, 1)	Activate lever 2, output "You operate the lever and hear a clunking noise."	Activate lever 2, output "You operate the lever and hear a clunking noise."	Pass
	"Use" button outside of rooms (3, 2) and (2, 1)	Output "Please enter a valid action"	Nothing	Fail
2.3)	"unlock" (Lower-case) command in room (1, 1)	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	"unlock" (Lower-case) command in room (1, 1) with key in inventory	Locked door opens, output "You hear a click. The door opens but the key breaks off in your hand.", key removed from inventory	Locked door opens, output "You hear a click. The door opens but the key breaks off in your hand.", key removed from inventory	Pass
	"unlock" (Lower-case) command outside of room (1, 1)	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	"Unlock" (Upper-case) command in room (1, 1)	Output "Please enter a valid action"	Nothing	Fail
	"Unlock" (Upper-case) command outside of room (1, 1)	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
2.4)	"hell" (Lower-case) command in room (2, 1)	Password door opens, output "The door creaks open. You seem to have entered the right password!"	Password door opens, output "The door creaks open. You seem to have entered the right password!"	Pass
	"hell" (Lower-case) command outside of room (2, 1)	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	"Hell" (Upper-case) command in room (2, 1)	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	"Hell" (Upper-case) command outside of room (2, 1)	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
2.5)	"Heal" button while at lower than 70 health	Add 30-40 to player health, output "You now have {Player health} hitpoints You swig a potion from your backpack and heal. Your health increased by {30-40} hitpoints"	Add 30-40 to player health, output "You now have 96 hitpoints You swig a potion from your backpack and heal. Your health increased by 31 hitpoints"	pass
	"Heal" button while at 70 health or higher	Set player health to 100, output "You now have 100 hitpoints. You swig a potion from your backpack and heal yourself to full health"	Set player health to 100, output "You now have 100 hitpoints. You swig a potion from your backpack and heal yourself to full health"	Pass

Test Case	ScenariExpected results	Actual results	Pass/Fail	Evidence (Optional)
2.6)	"Load" button	Set all variables to ones in text file, update stats, inventory and image, output "You have loaded a previous save."	Set all variables to ones in text file, update stats, inventory and image, output "You have loaded a previous save."	Pass
2.7)	"Save" button	Write all current variables to a text file, output "You have saved your progress."	Write all current variables to a text file, output "You have saved your progress."	Pass
2.8)	"Exit" button	Exit application	Exit application	Pass
2.9)	"Block" button in room (3, 2) while enemy is alive	Block, output "You have 100 hitpoints left You have blocked all damage that the monster did to you!" or "You have failed to block and the monster hit you for {block} hit points"	Block, output "You have 100 hitpoints left You have blocked all damage that the monster did to you!" or "You have failed to block and the monster hit you for {block} hit points"	Pass
	"Block" button in room (3, 2) while enemy is dead	Output "There is nothing to block here!"	Output "There is nothing to attack here!"	Fail
	"Block" button in room (1, 1) while enemy is alive	Block, output "You have 100 hitpoints left You have blocked all damage that the monster did to you!" or "You have failed to block and the monster hit you for {block} hit points"	Block, output "You have 100 hitpoints left You have blocked all damage that the monster did to you!" or "You have failed to block and the monster hit you for {block} hit points"	Pass
	"Block" button in room (1, 1) while enemy is dead	Output "There is nothing to block here!"	Output "There is nothing to attack here!"	Fail
	"Block" button in room (0, 1) while enemy is alive	Block, output "You have 100 hitpoints left You have blocked all damage that the monster did to you!" or "You have failed to block and the monster hit you for {block} hit points"	Block, output "You have 100 hitpoints left You have blocked all damage that the monster did to you!" or "You have failed to block and the monster hit you for {block} hit points"	Pass
	"Block" button in room (0, 1) while enemy is dead	Output "There is nothing to block here!"	Output "There is nothing to attack here!"	Fail
	"Block" button outside of rooms (3, 2), (1, 1) and (0, 1)	Output "There is nothing to block here!"	Output "There is nothing to attack here!"	Fail
2.10)	"Attack" button in room (3, 2) while enemy is alive	Attack, Output "Monster 1 has {activeMonsterHealth} hit points, you attack with your weapon"	Attack, Output "Monster 1 has {activeMonsterHealth} hit points, you attack with your weapon"	Pass

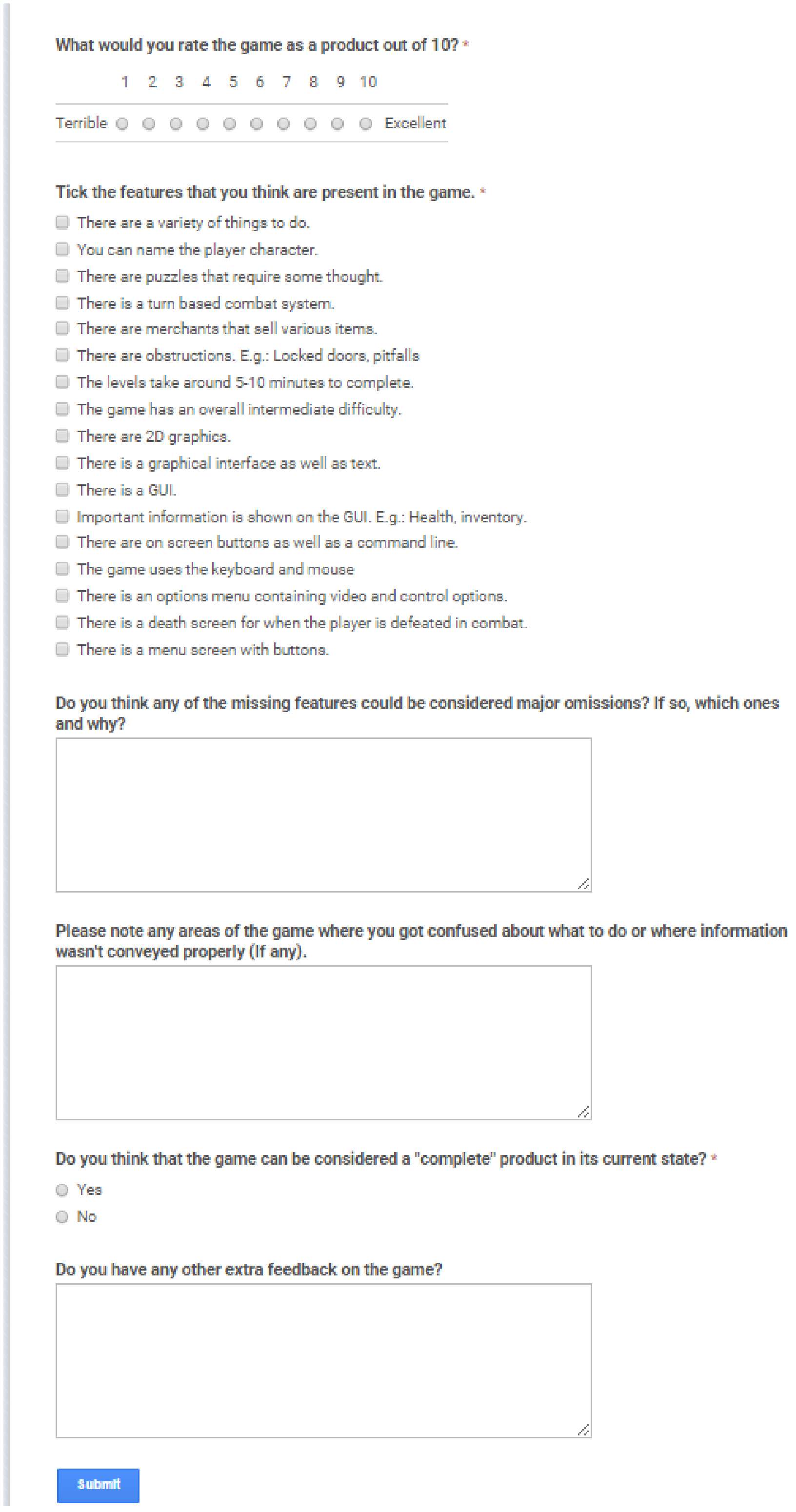
Test Case	ScenariExpected results	Actual results	Pass/Fail	Evidence (Optional)
	"Attack" button in room (3, 2) while enemy is dead	Output "The monster is already dead, you hack into its squishy dead body.... and get covered in goo!"	Output "There is nothing to attack here!"	Fail
	"Attack" button in room (1, 1) while enemy is alive	Attack, Output "Monster 1 has {activeMonsterHealth} hit points, you attack with your weapon"	Attack, Output "Monster 1 has {activeMonsterHealth} hit points, you attack with your weapon"	Pass
	"Attack" button in room (1, 1) while enemy is dead	Output "The monster is already dead, you hack into its squishy dead body.... and get covered in goo!"	Output "There is nothing to attack here!"	Fail
	"Attack" button in room (0, 1) while enemy is alive	Attack, Output "Monster 1 has {activeMonsterHealth} hit points, you attack with your weapon"	Attack, Output "Monster 1 has {activeMonsterHealth} hit points, you attack with your weapon"	Pass
	"Attack" button in room (0, 1) while enemy is dead	Output "The monster is already dead, you hack into its squishy dead body.... and get covered in goo!"	Output "There is nothing to attack here!"	Fail
	"Attack" button outside of rooms (3, 2), (1, 1) and (0, 1)	Output "There is nothing to attack here!"	Output "There is nothing to attack here!"	Pass
2.11)	"nothing" (Lower-case) command in room (2, 1) with enemy alive	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	"nothing" (Lower-case) command in room (2, 1) with enemy dead	Riddle door opens, Output "The door slides open. You answered the riddle correctly!"	Riddle door opens, Output "The door slides open. You answered the riddle correctly!"	Pass
	"nothing" (Lower-case) command outside room (2, 1)	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	"Nothing" (Upper-case) command in room (2, 1) with enemy alive	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	"Nothing" (Upper-case) command in room (2, 1) with enemy dead	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	"Nothing" (Upper-case) command outside room (2, 1)	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	"help" in any room	Output "forwards, left, right, back, collect, unlock"	Output "forwards, left, right, back, collect, unlock"	Pass
2.12)	"" in any room	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass
	"dsafnjw" in any room	Output "Please enter a valid action"	Output "Please enter a valid action"	Pass

Summary

The restrictions on movement are all in the correct places and work properly and the systems all work as they should, however there are a few minor problems regarding text outputs. Some commands in certain scenarios would not give the desired output but for most of these the output still worked adequately. Overall the system is rather solid technically but with some minor text output bugs.

User testing

For user testing I sent out a copy of the game to three end users along with a questionnaire. An image of the questionnaire and the results are shown below.



What would you rate the game as a product out of 10? *

1 2 3 4 5 6 7 8 9 10

Terrible ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ Excellent

Tick the features that you think are present in the game. *

- ☐ There are a variety of things to do.
- ☐ You can name the player character.
- ☐ There are puzzles that require some thought.
- ☐ There is a turn based combat system.
- ☐ There are merchants that sell various items.
- ☐ There are obstructions. E.g.: Locked doors, pitfalls
- ☐ The levels take around 5-10 minutes to complete.
- ☐ The game has an overall intermediate difficulty.
- ☐ There are 2D graphics.
- ☐ There is a graphical interface as well as text.
- ☐ There is a GUI.
- ☐ Important information is shown on the GUI. E.g.: Health, inventory.
- ☐ There are on screen buttons as well as a command line.
- ☐ The game uses the keyboard and mouse
- ☐ There is an options menu containing video and control options.
- ☐ There is a death screen for when the player is defeated in combat.
- ☐ There is a menu screen with buttons.

Do you think any of the missing features could be considered major omissions? If so, which ones and why?

Please note any areas of the game where you got confused about what to do or where information wasn't conveyed properly (If any).

Do you think that the game can be considered a "complete" product in its current state? *

○ Yes

○ No

Do you have any other extra feedback on the game?

What would you rate the game as a product out of 10?

1. 7
2. 7
3. 7

Tick the features that you think are present in the game.

1. There are a variety of things to do., There are puzzles that require some thought., There is a turn based combat system., There are obstructions. E.g.: Locked doors, pitfalls, The game has an overall intermediate difficulty., There are 2D graphics., There is a graphical interface as well as text., There is a GUI., Important information is shown on the GUI. E.g.: Health, inventory., There are on screen buttons as well as a command line., The game uses the keyboard and mouse
2. There are a variety of things to do., There are puzzles that require some thought., There is a turn based combat system., There are obstructions. E.g.: Locked doors, pitfalls, The game has an overall intermediate difficulty., There are 2D graphics., There is a graphical interface as well as text., There is a GUI., Important information is shown on the GUI. E.g.: Health, inventory., There are on screen buttons as well as a command line., The game uses the keyboard and mouse
3. There are a variety of things to do., There are puzzles that require some thought., There is a turn based combat system., The levels take around 5-10 minutes to complete., The game has an overall intermediate difficulty., There is a graphical interface as well as text., There is a GUI., Important information is shown on the GUI. E.g.: Health, inventory., The game uses the keyboard and mouse, There is a menu screen with buttons.

Do you think any of the missing features could be considered major omissions? If so, which ones and why?

1. Death screen to make the game look more polished.
2. There wasn't a health bar or similar which would make it easier for the user to understand their characters stats easier.

There could have been a tutorial or better help on how to play the game.

3. None.

Please note any areas of the game where you got confused about what to do or where information wasn't conveyed properly (if any).

1. The coloured doors with the letter 'P' marked on them. I did not know that I have to enter a password to get through them.
2. I didn't know how to pick up objects or how to open doors which could be because of little if any help tutorials.
3. None, it took quite a bit of thought though.

Do you think that the game can be considered a "complete" product in its current state?

1. No
2. No
3. Yes

Do you have any other extra feedback on the game?

1. Maybe make the help menu a bit clearer, at the moment it only shows a list of commands. It would be good if it was separated from the rest of the text e.g. -------[HELP]-------- and then another line ------------------- at the end with the commands inside it.
2. There is lacking story for the game or extra levels. It could also do with increased tutorial on how to play the game and make the list box clearer to understand.
3. Could have had further levels.

Summary

It seems as though my end users enjoy the game overall however there are some areas where information is poorly conveyed to the player. Judging from several responses it seem my efforts at showing the player how to play the game were not useful enough. "There could have been a tutorial or better help on how to play the game", "I didn't know how to pick up objects or how to open doors which could be because of little if any help tutorials" and "It could also do with increased tutorial on how to play the game and make the list box clearer to understand" show that users desired more adequate tutorials on how to play the game.

Of the 17 planned features, there were only 4 that all 3 users felt the game did not have at all. Those features are character naming, merchants, the options menu and the death screen. Judging from the answers to the other questions it seems as though no one particularly misses the options screen however one user said that there should be a "Death screen to make the game look more polished." One user also expressed that there should have been a health bar so that the player could see their health easier. "There wasn't a health bar or similar which would make it easier for the user to understand their characters stats easier." Two users expressed that they would have preferred further levels but as I have already explained I did not have enough time to implement more than one level. "Could have had further levels" and "There is lacking story for the game or extra levels" show this and also that one user would have preferred more emphasis on the story.

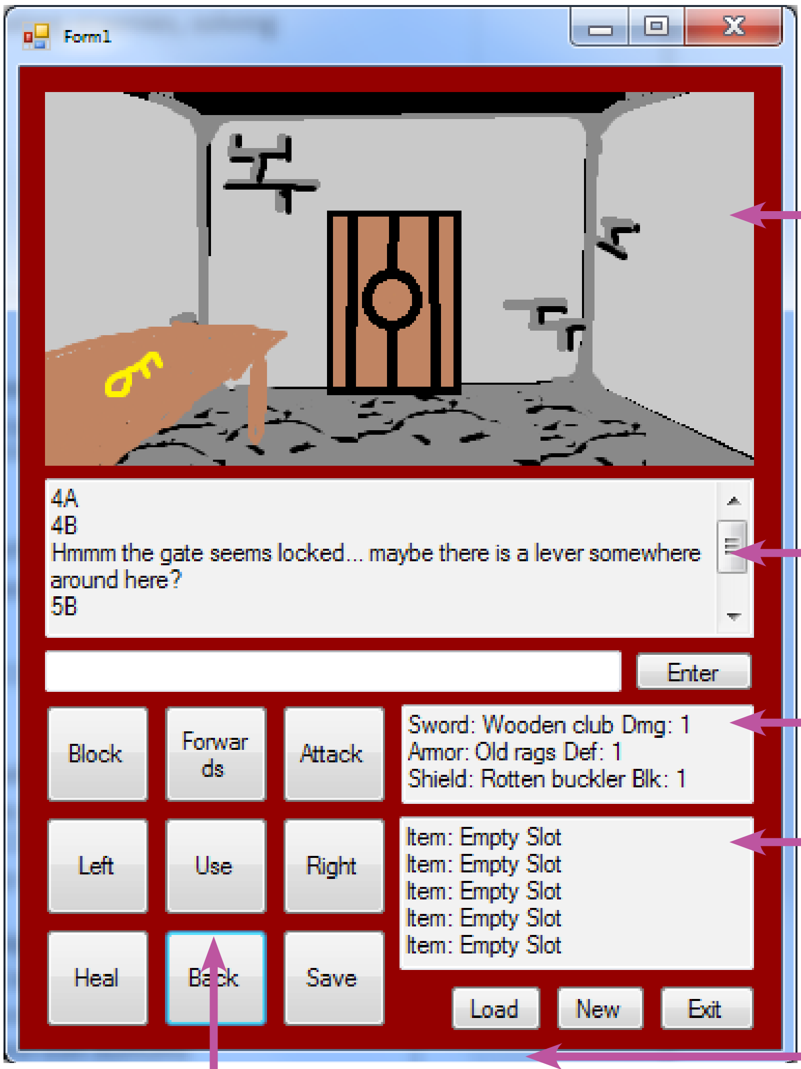
All three users rated the product as a 7 which I feel is a rather good score given the lack of many initial features, however two of the three users did not consider the game a "complete" product, probably due to the short length and lack of features.

Overall I think that the game was a success with my end user but only just. If I had the time to include the other planned features and provide sufficient help and tutorials I feel that my end users would have been happier with the product.

Section 4 – Documentation

Brief description

The goal of this game is to reach the exit of the dungeon by navigating the dark corridors while defeating enemies, solving puzzles and picking up loot.



4A
4B
Hmmm the gate seems locked... maybe there is a lever somewhere around here?
5B

Sword: Wooden club Dmg: 1
Armor: Old rags Def: 1
Shield: Rotten buckler Blk: 1

Item: Empty Slot
Item: Empty Slot
Item: Empty Slot
Item: Empty Slot
Item: Empty Slot

Image Output
This is the area where you will see an image of the room you are currently in.

Text output
This is where important information is displayed. It shows descriptions of actions and acts as a combat log.

Text Input
This is where you will input more detailed actions to interact with the game.

Stats and inventory output
These show your equipment stats and the items you currently hold in your inventory.

Save, Load, New and Exit buttons
These allow you to save your current progress, load a previous save, restart the game or exit the application without saving.

Action buttons
These buttons allow you to interact with the game. The buttons labelled with directions are used for movement. Attack and block are only used during combat. Heal allows you to use a healing item to regain some health.

Movement

Movement can be done by either using the directional buttons shown on the bottom left of the interface or by entering the direction you would like to move into the text input box shown in the middle of the interface. Movement is restricted to forwards, left, right and back and each moves you to an adjacent room in the direction you choose assuming there isn't a wall or locked door in the way.

Interaction

Interaction with the environment is done using the use button and the unlock and collect commands. The use button is used to activate levers, switches and other environmental interactions. The collect command is used to pick up items in the game. E.g.: Picking up a key from a table. Finally, the unlock command is used to unlock doors provided you have a key present in your inventory.

Combat

Upon encountering an enemy you have three options;

Attack the enemy for a random amount of damage.

Block for the chance to negate all incoming damage and attack with reduced damage.

Move to another room to leave combat. The enemy will have the same amount of health the next time you encounter it.

After either attacking or blocking the enemy will attack you for a random amount of damage. Once the enemy's health reaches below 0, the enemy will be defeated and the player may receive some loot. If the player is defeated the game must either be restarted or reloaded from a save file.

Installation

1. Place the game folder anywhere, preferably in a permanent space such as your user area.
2. Open up the solution and navigate to the areas of code where file locations are referenced. (Must have Visual Studio 2012 or later)
3. Change these file locations to where each file is located on your computer.
4. Click "Start" in the top left of the Visual Studio to play the game.

Troubleshooting/Q&A

Q: I can't pass through the locked door in room 7 even though I have a key.

A: You need to use the unlock command to open the door.

Q: Using the command "unlock" on the door in room 7 isn't working.

A: You must have a key present in your inventory to unlock the door. If you have a key and this still doesn't work, check your capitalisation and spelling.

Q: When I enter a command I get the message "Please enter a valid action".

A: The commands are case sensitive, only lower case is accepted. If that isn't the problem then check your spelling.

Q: I have pulled the lever but the gate in room 5 is still shut.

A: You must activate both levers before the gate will open.

Section 5 – Evaluation

Overall I feel that the project was a success despite the absence of several features I had planned in the designs. Here is a list of the initial planned features and whether they are present in the final version or not.

	Present in the final version (Yes, no, modified, removed)	
Planned Features	**Myself**	**End User**
Variety of things you can dYes	Yes	
To be able to name the character	NNo	
Puzzles that require some thought	Yes	Yes
To have a turn based combat system	Yes	Yes
Merchants that sell various helpful items	NNo	
Various obstructions such as locked doors and pitfalls	Yes	Yes
Levels that take around 5-10 minutes to complete	Yes	No
Have an overall intermediate difficulty	Yes	Yes
2D graphics	Yes	Yes
A graphical interface as well as text	Yes	Yes
A GUI to show information, buttons etc.	Yes	Yes
Information such as health, inventory displayed on the GUI	Yes	Yes
On screen buttons as well as a command line	Yes	Yes
Will use keyboard and mouse	Yes	Yes
An options menu containing video and control options	Removed	No
A death screen for when the player is defeated in combat	NNo	
A menu screen with buttons for new game, load game, options and exit	modified	No

The final version of the game only contained the first floor of the two that were planned due to time constraints. There was originally a main menu and pause menu but they were removed to save the time that would be wasted navigating menus. The buttons on the menus were relocated to the game screen where they were more accessible and useful. The options menu was also omitted due to the rearranging of buttons that were previously located on the main menu and pause menu. Having an options menu for two rather unnecessary options was also a waste of space. Character customisation in the way of the character's name and portrait were left out due to the player's character not being referenced in the text anyway and also due to the limited space on the form. Merchants and other dungeon dwellers were features that were going to be included on floor 2 however I ended up scrapping floor 2 so the feature was never developed. Implementing them into floor 1 would have required a complete re design so I decided to just omit the feature.

User reception was positive with the product receiving 7 out of 10 overall from the users I tested, however it seemed that most found the game to be an incomplete product despite the high score due to inadequate tutorials, some confusing concepts that were not fully explained such as the password door and riddle door and the absence of several features that were initially proposed. Overall I feel that the game was a success because despite the end users being underwhelmed by the user friendliness of the system, they overall still found the software to be rather solid in its own right.

If the software was ever to receive an update I would;

- Expand upon the first floor – Most of the end users felt that the game did not reach the target 5-10 minutes so making the first floor larger would fix this issue.
- Create more floors – The end users expressed that they would have preferred more floors. Adding the second floor and possibly more floors would fix this problem.
- Prioritise story and player choice – The game was originally supposed to have a larger emphasis on the story and choices the player made. The final version of the game is very much linear.
- Include character customisation – I wasn't able to add this due to several factors however if I was to place a higher priority on story and player choice then this would be a must.

Code

```
using System;
using System.IO;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace WindowsFormsApplication11
{
    public partial class Form1 : Form
    {

        //Whether the player can pass through the door.
        bool riddleDoor1 = false;
        bool passDoor1 = false;
        bool lockedDoor1 = false;

        //Room 3 2 Options
        bool lever32 = false;
        bool collect32 = false;

        //Room 3 0 Options
        bool collect30 = false;

        //Room 2 1 Options
        bool lever21 = false;

        //The equipment currenly being utilised by the player. these act as
multiplyers for their respective uses.
        int currentShieldValue = 1; //Block chance.
        int currentSwordValue = 1; //Sword damage.
        int currentArmourValue = 1; //Damage reduction.

        //Number of items in the players inventory.
        int keyNo = 0;
        int hPotionNo = 4;
        string shield = "Rotten buckler";
        string sword = "Wooden club";
        string armour = "Old rags";
        //Various player and enemy properties.
        bool enemy3Alive = true;
        bool enemy2Alive = true;
        bool enemy1Alive = true;
        int enemy3Health = 60;
        int enemy2Health = 30;
        int enemy1Health = 20;
        int playerHealth = 100;

        //Array to hold inventory
        string [] inventory = {"Empty Slot", "Empty Slot","Empty Slot","Empty
Slot","Empty Slot"} ;
        int inventoryLocation = 0;

        //Coordinate system.
        int playerX = 3;
        int playerY = 0;
        string[,] coordinates = new string[5, 5] { { "1A", "1B", "1C", "1D", "1E" }, {
"2A", "2B", "2C", "2D", "2E" }, { "3A", "3B", "3C", "3D", "3E" }, { "4A", "4B", "4C",
"4D", "4E" }, { "5A", "5B", "5C", "5D", "5E" }, };

        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            //Runs the code as soon as the form has loaded
            PlayerCurrentPos(playerX, playerY);
            statsUpdate();
            inventoryUpdate(inventory);
            TextOutput.Text = "For a list of common commands, enter \"Help\" in the
command line.";
        }

        public void PlayerAction(string command)
        {
            //This Method handles the next action based on what the player inputs.

            if (command == "forwards" && playerY != 4 && ((playerX == 3 && playerY ==
0) || (playerX == 3 && playerY == 1) || ((playerX == 4 && playerY == 1) && lever32 ==
true && lever21 == true) || ((playerX == 1 && playerY == 1) && lockedDoor1 == true)))
            {
                //Increment the players Y coordinate and display the players current
position.
                playerY++;
                TextOutput.Text = coordinates[playerX, playerY] + "\r\n" +
TextOutput.Text;
            }

            else if (command == "forwards" && ((playerX == 4 && playerY == 1) &&
(lever32 == false || lever21 == false)))
            {
                //Dsiplay "Gate Locked" message
                TextOutput.Text = "Hmmm the gate seems locked... maybe there is a
lever somewhere around here?" + "\r\n" + TextOutput.Text;
            }

            else if (command == "back" && playerY != 0 && ((playerX == 3 && playerY ==
1) || (playerX == 3 && playerY == 2) || (playerX == 4 && playerY == 2) || (playerX ==
1 && playerY == 2) || (playerX == 1 && playerY == 3)))
```

```
            {
                //Decrement the players Y coordinate and display the players current
position.
                playerY--;
                TextOutput.Text = coordinates[playerX, playerY] + "\r\n" +
TextOutput.Text;
            }

            else if (command == "left" && playerX != 0 && ((playerX == 0 && playerY ==
1) || ((playerX == 1 && playerY == 1) && riddleDoor1 == true) || ((playerX == 2 &&
playerY == 1) && passDoor1 == true) || (playerX == 3 && playerY == 1) || (playerX == 4
&& playerY == 1)))
            {
                //Decrement the players X coordinate and display the players current
position.
                playerX--;

                TextOutput.Text = coordinates[playerX, playerY] + "\r\n" +
TextOutput.Text;
            }

            else if (command == "right" && playerX != 4 && (((playerX == 3 && playerY
== 1) || (playerX == 2 && playerY == 1)) || (playerX == 1 && playerY == 1) || (playerX
== 0 && playerY == 1)))
            {
                //Increment the players X coordinate and display the players current
position.
                playerX++;
                TextOutput.Text = coordinates[playerX, playerY] + "\r\n" +
TextOutput.Text;
            }

            else if (command == "current")
            {
                //Display the players current coordinate.
                TextOutput.Text = TextOutput.Text + "\r\n" + coordinates[playerX,
playerY];
                TextOutput.Text = "Location: " + coordinates[playerX, playerY] +
"\r\n" + TextOutput.Text;
            }

            else if (command == "collect")
            {
                //Allows the player to pick up items and add them to their inventory
depending on the room they are in.
                if ((playerX == 3 && playerY == 2))
                {
                    if (enemy1Alive == true)
                    {
                        TextOutput.Text = "There is nothing to collect here... try
killing the monster!!" + "\r\n" + TextOutput.Text;
                    }

                    else if (enemy1Alive == false && collect32 == false)
                    {
                        currentSwordValue = currentSwordValue + 2;
                        sword = "Rusty Short Sword";
                        TextOutput.Text = "You collect a Rusty Short Sword and your
attack rating increases" + "\r\n" + TextOutput.Text;
                        TextOutput.Text = "You collect try to pick up food and store
it in your inventory" + "\r\n" + TextOutput.Text;
                        if (inventoryLocation < 4)
                        {
                            inventory[inventoryLocation] = "Cooked Meat";
                            inventoryLocation++;
                            TextOutput.Text = "You find some room in your bags and
keep the food" + "\r\n" + TextOutput.Text;
                            inventoryUpdate(inventory);
                            collect32 = true;
                            //Updated PICTURES to match
                        }
                        else
                        {
                            TextOutput.Text = "You cannot find any room in your bags."
+ "\r\n" + TextOutput.Text;
                        }

                    }
                    statsUpdate();



                }

                else if ((playerX == 3 && playerY == 0))
                {
                    if (collect30 == false)
                    {
                        if (inventoryLocation < 4)
                        {
                            inventory[inventoryLocation] = "Key";
                            inventoryLocation++;
                            TextOutput.Text = "You pick up a worn key off of a nearby
table." + "\r\n" + TextOutput.Text;
                            inventoryUpdate(inventory);
                            collect30 = true;
                        }
                        else
                        {
                            TextOutput.Text = "You cannot find any room in your bags."
+ "\r\n" + TextOutput.Text;
                        }
                    }
                    else
                    {
                        TextOutput.Text = "There is nothing to collect" + "\r\n" +
TextOutput.Text;
                    }
                }
            }

            else if (command == "hell" && (playerX == 2 && playerY == 1))
            {
                //Password for the password door
                TextOutput.Text = "The door creaks open. You seem to have entered the
right password!" + "\r\n" + TextOutput.Text;
                passDoor1 = true;
            }

            else if (command == "nothing" && (playerX == 1 && playerY == 1) &&
enemy2Alive == false && riddleDoor1 == false)
            {
                //Opens the riddle door if the enemy in the room is dead
                TextOutput.Text = "The door slides open. You answered the riddle
correctly!" + "\r\n" + TextOutput.Text;
                riddleDoor1 = true;
            }
            else if (command == "unlock" && (playerX == 1 && playerY == 1) &&
collect30 == true && lockedDoor1 == false && enemy2Alive == false)
            {
                //Unlocks the locked door if the player has a key in their inventory
and the enemy in the room is dead
                inventoryLocation--;
                inventory[inventoryLocation] = "Empty slot";
                TextOutput.Text = "You hear a click. The door opens but the key breaks
off in your hand." + "\r\n" + TextOutput.Text;
                lockedDoor1 = true;
            }

            else if (command == "help")
            {
                //Lists the common commands in the text output

                TextOutput.Text = "forwards" + "\r\n" + "left" + "\r\n" + "right" +
"\r\n" + "back" + "\r\n" + "current" + "\r\n" + "collect" + "\r\n" + "unlock" + "\r\n"
+ TextOutput.Text;
            }

            else
            {
                TextOutput.Text = "Please enter a valid action." + "\r\n" +
TextOutput.Text;
            }



        }

        public void EnterText_Click(object sender, EventArgs e)
        {
            string input = "";
            //On button press, assign text currently entered in the input box to input
variable.
            input = TextInput.Text;
            PlayerAction(input);
            PlayerCurrentPos(playerX, playerY);
            //Reset input
            input = null;
        }

        public void PlayerCurrentPos(int x, int y)
        {
            //Retreive current player coordinates, output an image of the current room
and set properties.
            // -----------------------------------------------------------------------
-------------------------------------
            // Room 3 0 Picture Display Choices
            // -----------------------------------------------------------------------
-------------------------------------
            if ((x == 3 && y == 0) && collect30 == false)
            {
                GameDisplay.Image = new Bitmap(@"E:\School\NEWEST\Computing\F454\F454
Game HOME VER\WindowsFormsApplication11\Resources\Room1 KEY.png");
            }
            else if ((x == 3 && y == 0) && collect30 == true)
            {
                GameDisplay.Image = new Bitmap(@"E:\School\NEWEST\Computing\F454\F454
Game HOME VER\WindowsFormsApplication11\Resources\Room1 Empty.png");
            }

            // -----------------------------------------------------------------------
-------------------------------------
```

```
            // Room 3 1 Picture Display Choices
            // -----------------------------------------------------------------------
-------------------------------------
            if (x == 3 && y == 1)
            {
                GameDisplay.Image = new Bitmap(@"E:\School\NEWEST\Computing\F454\F454
Game HOME VER\WindowsFormsApplication11\Resources\Room2.png");
            }

            // -----------------------------------------------------------------------
-------------------------------------
            // Room 2 1 Picture Display Choices
            // -----------------------------------------------------------------------
-------------------------------------
            if (x == 2 && y == 1)
            {
                GameDisplay.Image = new Bitmap(@"E:\School\NEWEST\Computing\F454\F454
Game HOME VER\WindowsFormsApplication11\Resources\Room3.png");
            }

            // -----------------------------------------------------------------------
-------------------------------------
            // Room 4 1 Picture Display Choices
            // -----------------------------------------------------------------------
-------------------------------------
            if (x == 4 && y == 1)
            {
                GameDisplay.Image = new Bitmap(@"E:\School\NEWEST\Computing\F454\F454
Game HOME VER\WindowsFormsApplication11\Resources\Room5.png");
            }

            // -----------------------------------------------------------------------
-------------------------------------
            // Room 3 2 Picture Display Choices
            // -----------------------------------------------------------------------
-------------------------------------
            if (x == 3 && y == 2 && enemy1Alive == true)
            {
                TextOutput.Text = TextOutput.Text + "\r\n" + "You encounter an enemy!!
Press Attack/Heal/Block to fight or move from the room to quit combat";
                GameDisplay.Image = new Bitmap(@"E:\School\NEWEST\Computing\F454\F454
Game HOME VER\WindowsFormsApplication11\Resources\Room 4 Monster.png");
            }
            else if (x == 3 && y == 2 && enemy1Alive == false && collect32==false)
            {
                GameDisplay.Image = new Bitmap(@"E:\School\NEWEST\Computing\F454\F454
Game HOME VER\WindowsFormsApplication11\Resources\Room 4 meat and sword.png");
            }
            else if (x == 3 && y == 2 && enemy1Alive == false && collect32 == true)
            {
                GameDisplay.Image = new Bitmap(@"E:\School\NEWEST\Computing\F454\F454
Game HOME VER\WindowsFormsApplication11\Resources\Room 4 empty.png");
            }

            // -----------------------------------------------------------------------
-------------------------------------
            // Room 4 2 Picture Display Choices
            // -----------------------------------------------------------------------
-------------------------------------
            if (x == 4 && y == 2)
            {
                GameDisplay.Image = new Bitmap(@"E:\School\NEWEST\Computing\F454\F454
Game HOME VER\WindowsFormsApplication11\Resources\Room 6.png");
            }
            // -----------------------------------------------------------------------
------------------------------------
            // Room 1 1 Picture Display Choices
            // -----------------------------------------------------------------------
------------------------------------
            if (x == 1 && y == 1 && enemy2Alive == true)
            {
                GameDisplay.Image = new Bitmap(@"E:\School\NEWEST\Computing\F454\F454
Game HOME VER\WindowsFormsApplication11\Resources\Room  7 Monster.png");

            }
            else if (x == 1 && y == 1 && enemy2Alive == false)
            {
                GameDisplay.Image = new Bitmap(@"E:\School\NEWEST\Computing\F454\F454
Game HOME VER\WindowsFormsApplication11\Resources\Room  7 empty.png");
            }

            if ((x == 1 && y == 1) && enemy2Alive == false && riddleDoor1 == false)
            {
                TextOutput.Text = "The poor have it. The rich need it. If you eat it,
you will die. what is it?" + "\r\n" + TextOutput.Text;
            }

            // -----------------------------------------------------------------------
------------------------------------
            // Room 0 1 Picture Display Choices
            // -----------------------------------------------------------------------
------------------------------------
            if (x == 0 && y == 1 && enemy3Alive == true)
            {
                GameDisplay.Image = new Bitmap(@"E:\School\NEWEST\Computing\F454\F454
Game HOME VER\WindowsFormsApplication11\Resources\Room 8 empty monster.png");
            }
            else if (x == 0 && y == 1 && enemy3Alive == false)
            {
                GameDisplay.Image = new Bitmap(@"E:\School\NEWEST\Computing\F454\F454
Game HOME VER\WindowsFormsApplication11\Resources\Room 8 empty.png");
            }

            // -----------------------------------------------------------------------
------------------------------------
            // Room 1 2 Picture Display Choices
            // -----------------------------------------------------------------------
------------------------------------
            if (x == 1 && y == 2)
            {
                GameDisplay.Image = new Bitmap(@"E:\School\NEWEST\Computing\F454\F454
Game HOME VER\WindowsFormsApplication11\Resources\Room 9.png");
                MessageBox.Show("You have escaped the dungeon!");
                Application.Exit();
            }

        }

        public void combatSystem(string action)
        {
            //Combat actions for room (3, 2).
            if (action == "attack" && playerX == 3 && playerY == 2 && enemy1Alive ==
true)
            {
                enemy1Health = combatAttack(enemy1Health);
                if (enemy1Health <= 0)
                {
                    enemy1Alive = false;
                    GameDisplay.Image = new
Bitmap(@"E:\School\NEWEST\Computing\F454\F454 Game HOME
VER\WindowsFormsApplication11\Resources\Room 4 meat and sword.png");
                }
            }

            else if (action == "block" && playerX == 3 && playerY == 2 && enemy1Alive
== true)
            {
                enemy1Health = combatBlock(enemy1Health);
                if (enemy1Health <= 0)
                {
                    enemy1Alive = false;
                    GameDisplay.Image = new
Bitmap(@"E:\School\NEWEST\Computing\F454\F454 Game HOME
VER\WindowsFormsApplication11\Resources\Room 4 meat and sword.png");
                }
            }

            //Combat actions for (1, 1).
            else if (action == "attack" && playerX == 1 && playerY == 1 && enemy2Alive
== true)
            {
                enemy2Health = combatAttack(enemy2Health);
                if (enemy2Health <= 0)
                {
                    enemy2Alive = false;
                    GameDisplay.Image = new
Bitmap(@"E:\School\NEWEST\Computing\F454\F454 Game HOME
VER\WindowsFormsApplication11\Resources\Room  7 empty.png");
                    TextOutput.Text = "The monster dropped a suit of Stone Armour!" +
"\r\n" + TextOutput.Text;
                    armour = "Stone Armour";
                    currentArmourValue = currentArmourValue + 2;
                    statsUpdate();
                }
            }

            else if (action == "block" && playerX == 1 && playerY == 1 && enemy2Alive
== true)
            {
                enemy2Health = combatBlock(enemy2Health);
                if (enemy2Health <= 0)
                {
                    enemy2Alive = false;
                    GameDisplay.Image = new
Bitmap(@"E:\School\NEWEST\Computing\F454\F454 Game HOME
VER\WindowsFormsApplication11\Resources\Room  7 empty.png");
                    TextOutput.Text = "The monster dropped a suit of Stone Armour!" +
"\r\n" + TextOutput.Text;
                    armour = "Stone Armour";
                    currentArmourValue = currentArmourValue + 2;
                    statsUpdate();
                }
            }

            //Combat actions for (0, 1).
            else if (action == "attack" && playerX == 0 && playerY == 1 && enemy3Alive
== true)
            {
                enemy3Health = combatAttack(enemy3Health);
                if (enemy3Health <= 0)
                {
                    enemy3Alive = false;
                    GameDisplay.Image = new
Bitmap(@"E:\School\NEWEST\Computing\F454\F454 Game HOME
VER\WindowsFormsApplication11\Resources\Room 8 empty.png");
                    TextOutput.Text = "The monster dropped a Beaten Shield!" + "\r\n"
+ TextOutput.Text;
                    shield = "Beaten Shield";
                    currentShieldValue = currentShieldValue + 2;
                    statsUpdate();
                }
            }

            else if (action == "block" && playerX == 0 && playerY == 1 && enemy3Alive
== true)
            {
                enemy3Health = combatBlock(enemy3Health);
                if (enemy3Health <= 0)
                {
                    enemy3Alive = false;
                    GameDisplay.Image = new
Bitmap(@"E:\School\NEWEST\Computing\F454\F454 Game HOME
VER\WindowsFormsApplication11\Resources\Room 8 empty.png");
                    TextOutput.Text = "The monster dropped a Beaten Shield!" + "\r\n"
+ TextOutput.Text;
                    shield = "Beaten Shield";
                    currentShieldValue = currentShieldValue + 2;
                    statsUpdate();
                }
            }

            else
            {
                TextOutput.Text = "There is nothing to attack here!" + "\r\n" +
TextOutput.Text;
            }

        }

        private void ForwardsButton_Click(object sender, EventArgs e)
        {
            //Input "forwards".
            PlayerAction("forwards");
            PlayerCurrentPos(playerX, playerY);
        }

        private void LeftButton_Click(object sender, EventArgs e)
        {
            //Input "left".
            PlayerAction("left");
            PlayerCurrentPos(playerX, playerY);
        }

        private void RightButton_Click(object sender, EventArgs e)
        {
            //Input "right".
            PlayerAction("right");
            PlayerCurrentPos(playerX, playerY);
        }

        private void BackButton_Click(object sender, EventArgs e)
        {
            //Input "back".
            PlayerAction("back");
            PlayerCurrentPos(playerX, playerY);
        }
        private void TextOutput_TextChanged(object sender, EventArgs e)
        {

        }

        private void TextInput_TextChanged(object sender, EventArgs e)
        {

        }

        private void GameDisplay_Click(object sender, EventArgs e)
        {

        }

        private void AttackButton_Click(object sender, EventArgs e)
        {
            //Input attack into combat method
            string action = "attack";

            combatSystem(action);
        }

        private int combatAttack(int activeMonsterHealth)
        {
            Random chance = new Random();

            if (activeMonsterHealth > 0)
            {
                // Player attacks monster
                TextOutput.Text = "Monster 1 has " + activeMonsterHealth + " hit 
points, you attack with your weapon" + "\r\n" + TextOutput.Text;
                activeMonsterHealth = activeMonsterHealth - chance.Next(5 + 
currentSwordValue, 10 + currentSwordValue);
                //Monster attacks player
                TextOutput.Text = "You have " + playerHealth + " hit points.. the 
monster attacks you!" + "\r\n" + TextOutput.Text;
                playerHealth = playerHealth - chance.Next(4, 7);

                //Output results
                TextOutput.Text = "You have " + playerHealth + " hitpoints left" + 
"\r\n" + TextOutput.Text;
                if (activeMonsterHealth <= 0)
                {
                    TextOutput.Text = "The monster has died" + "\r\n" + 
TextOutput.Text;
                }
                else
                {
                    TextOutput.Text = "The monster has " + activeMonsterHealth + " 
hitpoints left" + "\r\n" + TextOutput.Text;
                }
            }
            else if (playerHealth <= 0)
            {
                TextOutput.Text = "Hahahahha you died!" + "\r\n" + TextOutput.Text;
            }

            else if (activeMonsterHealth <= 0)
            {
                TextOutput.Text = "The monster is already dead, you hack into it's 
squishy dead body.... and get covered in goo!" + "\r\n" + TextOutput.Text;
            }

            return activeMonsterHealth;
        }

        private int combatBlock(int activeMonsterHealth)
        {
            Random chance = new Random();
            int block;

            if (activeMonsterHealth > 0)
            {
                //Player attacks monster
                TextOutput.Text = "Monster 1 has " + activeMonsterHealth + " hit 
points, you set a defensive stance" + "\r\n" + TextOutput.Text;
                activeMonsterHealth = (int)((double)activeMonsterHealth - 
(double)chance.Next(5, 10) * 0.5);
                //Monster attacks player
                TextOutput.Text = "You have " + playerHealth + " hit points.. the 
monster attacks you!" + "\r\n" + TextOutput.Text;

                block = chance.Next(4, 7) * chance.Next(0,1);
                playerHealth = playerHealth - block;
                if (block == 0)
                {
                    TextOutput.Text = "You have blocked all damage that the monster 
did to you!" + "\r\n" + TextOutput.Text;
                }
                else
                {
                    TextOutput.Text = "You have failed to block and the monster hit 
you for" + block + "hit points" + "\r\n" + TextOutput.Text;
                }
                //Output results
                TextOutput.Text = "You have " + playerHealth + " hitpoints left" + 
"\r\n" + TextOutput.Text;
                if (activeMonsterHealth <= 0)
                {
                    TextOutput.Text = "The monster has died" + "\r\n" + 
TextOutput.Text;
                }
                else
                {
                    TextOutput.Text = "The monster has " + activeMonsterHealth + " 
hitpoints left" + "\r\n" + TextOutput.Text;
                }
            }
            else if (playerHealth <= 0)
            {
                TextOutput.Text = "Hahahahha you died!" + "\r\n" + TextOutput.Text;
            }

            else if (activeMonsterHealth <= 0)
            {
                TextOutput.Text = "The monster is already dead, you hack into it's 
squishy dead body.... and get covered in goo!" + "\r\n" + TextOutput.Text;
            }

            return activeMonsterHealth;
        }

        private void BlockButton_Click(object sender, EventArgs e)
        {
            //Input block into combat method
            string action = "block";
            combatSystem(action);
        }

        private void HealButton_Click(object sender, EventArgs e)
        {
            //Heals the player for a random amount between 30 and 40
            if (hPotionNo > 0)
            {
                Random heal = new Random();
                int healthBoost = heal.Next(30, 40);
                hPotionNo--;
                playerHealth = playerHealth + healthBoost;

                if (playerHealth <= 100)
                {
                    TextOutput.Text = "You swig a potion from your backpack and heal.
Your health increased by " + healthBoost + " hitpoints" + "\r\n" + TextOutput.Text;
                    TextOutput.Text = "You now have " + playerHealth + " hitpoints" +
"\r\n" + TextOutput.Text;
                }

                else
                {
                    playerHealth = 100;
                    TextOutput.Text = "You swig a potion from your backpack and heal
yourself to full health" + "\r\n" + TextOutput.Text;
                    TextOutput.Text = "You now have " + playerHealth + " hitpoints" +
"\r\n" + TextOutput.Text;
                }
            }

            else
            {
                TextOutput.Text = "You have no heal potions and stay at " +
playerHealth + " hitpoints" + "\r\n" + TextOutput.Text;
            }
        }

        private void OKButton_Click(object sender, EventArgs e)
        {
            //Activates environmental objects depending on player position
            if (playerX == 3 && playerY == 2)
            {
                if (lever32 == false && enemy1Alive == false)
                {
                    TextOutput.Text = "You operate the lever and hear a clunking
noise." + "\r\n" + TextOutput.Text;
                    lever32 = true;
                }
                else if (lever32 == false && enemy1Alive == true)
                {
                    TextOutput.Text = "The lever is being guarded.. fight to the
death!" + "\r\n" + TextOutput.Text;
                    lever32 = false;
                }
                else if (lever32 == true)
                {
                    TextOutput.Text = "It appears that you have already used this
lever!" + "\r\n" + TextOutput.Text;
                }
            }
            if (playerX == 2 && playerY == 1)
            {
                if (lever21 == false)
                {
                    TextOutput.Text = "You operate the lever and hear a clunking
noise." + "\r\n" + TextOutput.Text;
                    lever21 = true;
                }
                else if (lever21 == true)
                {
                    TextOutput.Text = "It appears that you have already used this
lever!" + "\r\n" + TextOutput.Text;
                }
            }
        }

        private void inventoryUpdate(string[] inventory)
        {
            //Updates the inventory box with the most recent values
            InventoryBox.Text = "";
            for (int i = 0; i < 5; i++)
            {
                InventoryBox.Text = InventoryBox.Text + "Item: " + inventory[i] +
"\r\n" ;
            }
        }

        private void statsUpdate()
        {
            //Updates the stat box with the most recent values
            StatBox.Text = "";
            StatBox.Text = StatBox.Text + "Sword: " + sword + " Dmg: " +
currentSwordValue + "\r\n";
            StatBox.Text = StatBox.Text + "Armor: " + armour + " Def: " +
currentArmourValue + "\r\n";
            StatBox.Text = StatBox.Text + "Shield: " + shield + " Blk: " +
currentShieldValue + "\r\n";
        }

        private void NewButton_Click(object sender, EventArgs e)
        {
            //Restarts the application.
            Application.Restart();
        }

        private void ExitButton_Click(object sender, EventArgs e)
        {
            //Closes application.
            Application.Exit();
        }

        private void LoadButton_Click(object sender, EventArgs e)
        {
            //On button click, loads the values stored in the text file onto global
variables.
            using (StreamReader loadGame = new StreamReader(@"C:\Save.txt"))
            {
                riddleDoor1 = Convert.ToBoolean(loadGame.ReadLine());
                passDoor1 = Convert.ToBoolean(loadGame.ReadLine());
                lockedDoor1 = Convert.ToBoolean(loadGame.ReadLine());
                lever32 = Convert.ToBoolean(loadGame.ReadLine());
                collect32 = Convert.ToBoolean(loadGame.ReadLine());
                collect30 = Convert.ToBoolean(loadGame.ReadLine());
                lever21 = Convert.ToBoolean(loadGame.ReadLine());
                currentShieldValue = Convert.ToInt16(loadGame.ReadLine());
                currentSwordValue = Convert.ToInt16(loadGame.ReadLine());
                currentArmourValue = Convert.ToInt16(loadGame.ReadLine());
                keyNo = Convert.ToInt16(loadGame.ReadLine());
                hPotionNo = Convert.ToInt16(loadGame.ReadLine());
                shield = loadGame.ReadLine();
                sword = loadGame.ReadLine();
                armour = loadGame.ReadLine();
                enemy3Alive = Convert.ToBoolean(loadGame.ReadLine());
                enemy2Alive = Convert.ToBoolean(loadGame.ReadLine());
                enemy1Alive = Convert.ToBoolean(loadGame.ReadLine());
                enemy3Health = Convert.ToInt16(loadGame.ReadLine());
                enemy2Health = Convert.ToInt16(loadGame.ReadLine());
                enemy1Health = Convert.ToInt16(loadGame.ReadLine());
                playerHealth = Convert.ToInt16(loadGame.ReadLine());
                inventory[0] = loadGame.ReadLine();
                inventory[1] = loadGame.ReadLine();
                inventory[2] = loadGame.ReadLine();
                inventory[3] = loadGame.ReadLine();
                inventory[4] = loadGame.ReadLine();
                inventoryLocation = Convert.ToInt16(loadGame.ReadLine());
                playerX = Convert.ToInt16(loadGame.ReadLine());
                playerY = Convert.ToInt16(loadGame.ReadLine());
                statsUpdate();
                inventoryUpdate(inventory);
                PlayerCurrentPos(playerX, playerY);
                TextOutput.Text = "You have loaded a previous save." + "\r\n" +
TextOutput.Text;
            }

        }
        private void MenuButton_Click(object sender, EventArgs e)
        {
            //On button click, writes current values of all global variables to a text
file.
            using (StreamWriter saveGame = new StreamWriter(@"C:\Save.txt"))
            {
                saveGame.WriteLine(riddleDoor1);
                saveGame.WriteLine(passDoor1);
                saveGame.WriteLine(lockedDoor1);
                saveGame.WriteLine(lever32);
                saveGame.WriteLine(collect32);
                saveGame.WriteLine(collect30);
                saveGame.WriteLine(lever21);
                saveGame.WriteLine(currentShieldValue);
                saveGame.WriteLine(currentSwordValue);
                saveGame.WriteLine(currentArmourValue);
                saveGame.WriteLine(keyNo);
                saveGame.WriteLine(hPotionNo);
                saveGame.WriteLine(shield);
                saveGame.WriteLine(sword);
                saveGame.WriteLine(armour);
                saveGame.WriteLine(enemy3Alive);
                saveGame.WriteLine(enemy2Alive);
                saveGame.WriteLine(enemy1Alive);
                saveGame.WriteLine(enemy3Health);
                saveGame.WriteLine(enemy2Health);
                saveGame.WriteLine(enemy1Health);
                saveGame.WriteLine(playerHealth);
                saveGame.WriteLine(inventory[0]);
                saveGame.WriteLine(inventory[1]);
                saveGame.WriteLine(inventory[2]);
                saveGame.WriteLine(inventory[3]);
                saveGame.WriteLine(inventory[4]);
                saveGame.WriteLine(inventoryLocation);
                saveGame.WriteLine(playerX);
                saveGame.WriteLine(playerY);
                saveGame.Close();
                TextOutput.Text = "You have saved your progress." + "\r\n" +
TextOutput.Text;
            }

        }
    }
}
```

The small print

👍 👎

We'd like to know your view on the resources we produce. By clicking on the 'Like' or 'Dislike' button you can help us to ensure that our resources work for you. When the email template pops up please add additional comments if you wish and then just click 'Send'. Thank you.

If you do not currently offer this OCR qualification but would like to do so, please complete the Expression of Interest Form which can be found here: www.ocr.org.uk/expression-of-interest

We will inform centres about any changes to the specification. We will also publish changes on our website. The latest version of our specification will always be the one on our website (www.ocr.org.uk) and this may differ from printed versions.

ocr.org.uk/alevelreform

OCR customer contact centre

General qualifications

Telephone 01223 553998
Facsimile 01223 552627
Email general.qualifications@ocr.org.uk

A DIVISION OF CAMBRIDGE ASSESSMENT

LLOYD'S REGISTER · LRQA
ISO 9001

UKAS MANAGEMENT SYSTEMS
001