Ceebot and C# Console Independent Study and Project Log book

Name Robert Collcott

ID 21302939

Course Computing

Lecturer Richard Jones

Module Title and Code Programming Concepts CO452

Session Time 1PM – 4PM – Thursdays

Contents

Table of Contents

[Unit 1 Introduction to Ceebot - 2 Independent Study Tasks 5](#_Toc410335186)

[Unit 1 Independent Study Exercise 1.7 [Killer Wasps] 5](#_Toc410335187)

[Unit 1 Independent Study - Exercise 2 - Task 3.4 Production Chain 8](#_Toc410335188)

[Unit 2 Input and Output - 3 Independent Study Tasks 12](#_Toc410335189)

[Unit 2 Independent Study - Task 1 - A modification of the draw rectangles program 12](#_Toc410335190)

[13](#_Toc410335191)

[13](#_Toc410335192)

[13](#_Toc410335194)

[Unit 2 Independent Study - Task 2 Target Buster 2 – 5.2 14](#_Toc410335195)

[Unit 2 Independent Study - Task 3 5.6 Target Destruction 15](#_Toc410335196)

[Unit 3 Independent Study - Tasks 15](#_Toc410335197)

[8.2 Shoot Out 16](#_Toc410335198)

[8.6 Loop the Loop 17](#_Toc410335199)

[17](#_Toc410335200)

[17](#_Toc410335202)

[18](#_Toc410335204)

[Unit 3 Independent Study Task 3 - 11.3 Shooting Practise 18](#_Toc410335205)

[Unit 4 Selection Independent Study Tasks 20](#_Toc410335206)

[9.6 Alien Destruction 20](#_Toc410335207)

[5.5 A further modification of the draw variables rectangle program 21](#_Toc410335208)

[9.6 Alien Destruction 23](#_Toc410335209)

[Unit 5 Loops part 2 Independent Study 3 Tasks 25](#_Toc410335210)

[12.7 Ant Attack 25](#_Toc410335211)

[15.1 Zig Zag 26](#_Toc410335212)

[14.7 Autopilot 27](#_Toc410335213)

[28](#_Toc410335214)

[Unit 6 Functions - 2 Independent Study Tasks 29](#_Toc410335215)

[19.3 One More Flower 29](#_Toc410335216)

[19.4 Triangle function 31](#_Toc410335217)

[19.4 Triangle function extra 32](#_Toc410335218)

[Unit 7 Functions and Parameters - Independent Study Tasks 34](#_Toc410335219)

[21.6 The Fly to Function 34](#_Toc410335220)

[21.6 The Fly to Function extra 35](#_Toc410335221)

[36](#_Toc410335222)

[36](#_Toc410335224)

[Unit 8 - Arrays – Independent Study tasks 37](#_Toc410335225)

[Unit 8 Arrays exercises conversion to use functions 37](#_Toc410335226)

[Unit 9 Ceebot Project --- 38](#_Toc410335227)

[Project Extension Work -- Part 1 42](#_Toc410335228)

[Project Extension Work -- Part 2 – Autopilot 43](#_Toc410335229)

[Final piece of project - extension work 3 - finish off the enemy 44](#_Toc410335230)

[Unit 1 C# Console – Input and Output 45](#_Toc410335231)

[1.6 - Rocket Game - Independent Study 45](#_Toc410335232)

[1.7 Planetary Sums - Independent Study 47](#_Toc410335233)

[Unit 2 C# Console - Designing programs 49](#_Toc410335234)

[2.4 Food for thought independent study task Unit 2 C# Console 50](#_Toc410335235)

[Source Code 52](#_Toc410335251)

[Unit 3 C# Console Sequence Selection and Iteration 52](#_Toc410335252)

[Unit 3.6 Game On 52](#_Toc410335253)

[55](#_Toc410335254)

[55](#_Toc410335255)

[Unit 3.7 Game choices 56](#_Toc410335256)

[Unit 4 C# Console Classes Objects and Methods 58](#_Toc410335257)

[4.6 Nuclear Control 58](#_Toc410335258)

[4.7 More Craps!! 59](#_Toc410335259)

[4.6 Nuclear Control 59](#_Toc410335260)

[4.7 More Craps 60](#_Toc410335261)

## Unit 1 Introduction to Ceebot - 2 Independent Study Tasks

## 

## Unit 1 Independent Study Exercise 1.7 [Killer Wasps]

This complex task was to get the TNT cubes onto a spaceship without being hit by dangerous plants and wasps breaking your shield. To start off with is the algorithm for the program

**Algorithm**

**Declare Variables**

move 50 meters forward

turn 20 degrees to the right

move 30 meters forward

turn 30 degrees left

move 40 meters

grab the object

turn 160 degrees

move 110 meters

drop the object onto the spaceship

**Source Code**

extern void object::Task5\_3()

{

// Robert Collcott

// ID 21302939

// Course Computing

// Week 2 Exercise 1.7

// 9th October 2014

float distance1;

float angle1;

float distance2;

float angle2;

float distance3;

float distance4

distance1 = 50;

distance2 = 30

distance3 = 40

distance4 = 110

angle1 = -20

angle2 = 30

angle3 = 160

move(distance1);

turn(angle1);

move(distance2);

turn(angle2);

move(distance3);

grab();

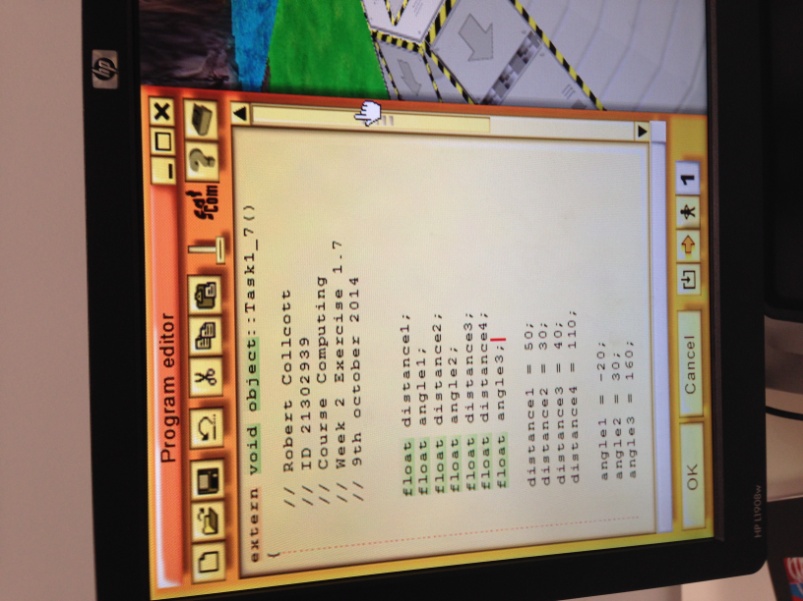
turn(angle3);

move(distance4);

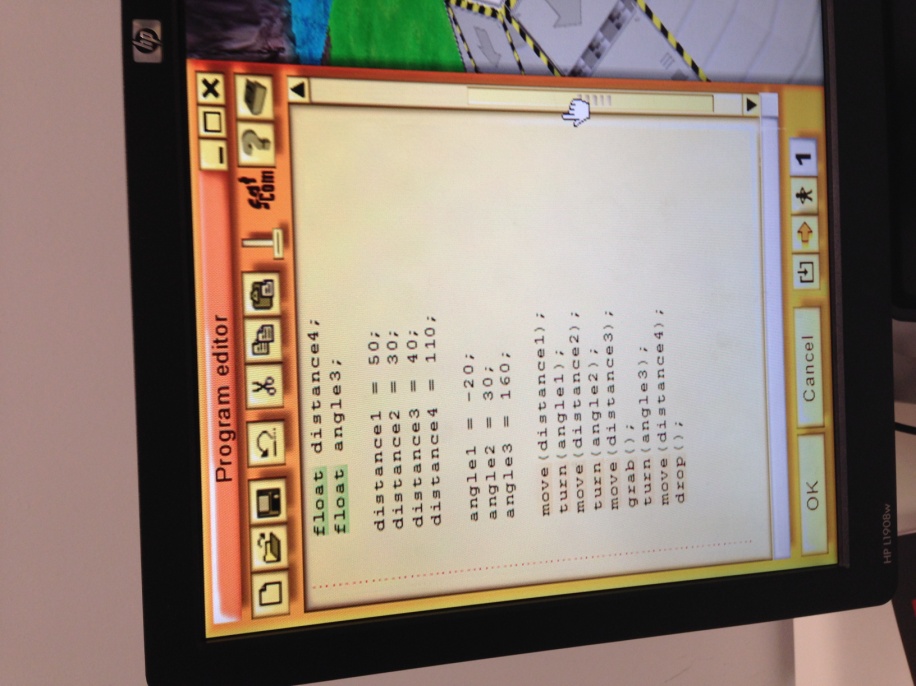
drop();

}

As you can see here I am using variables to help with the distance and direction

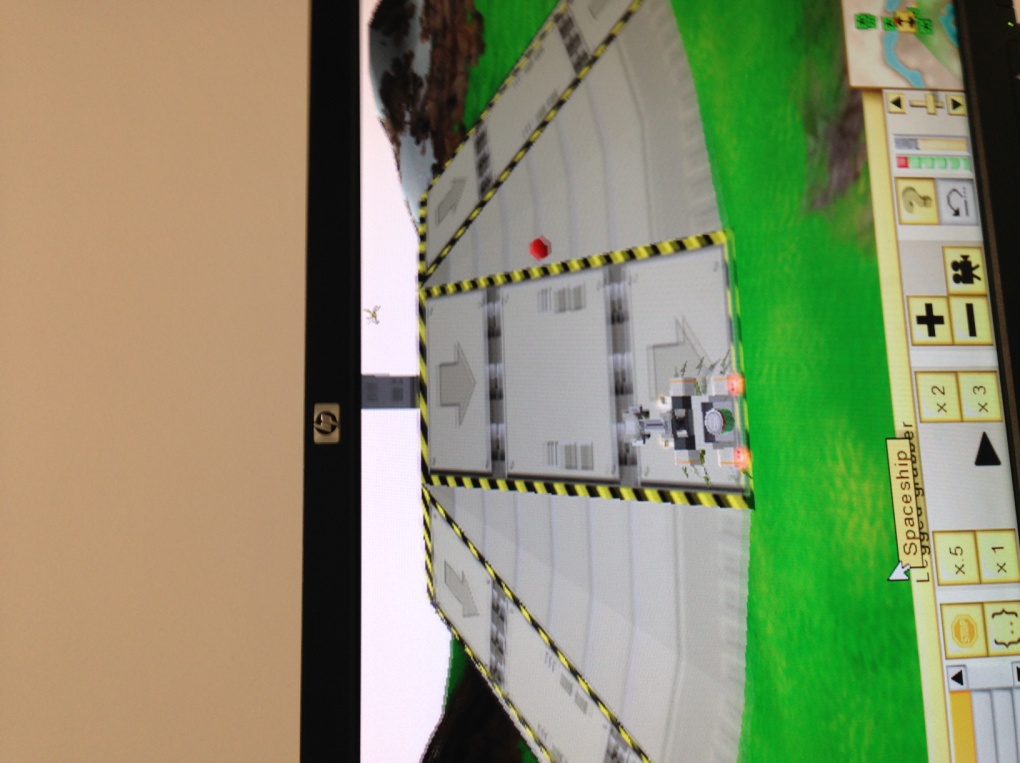


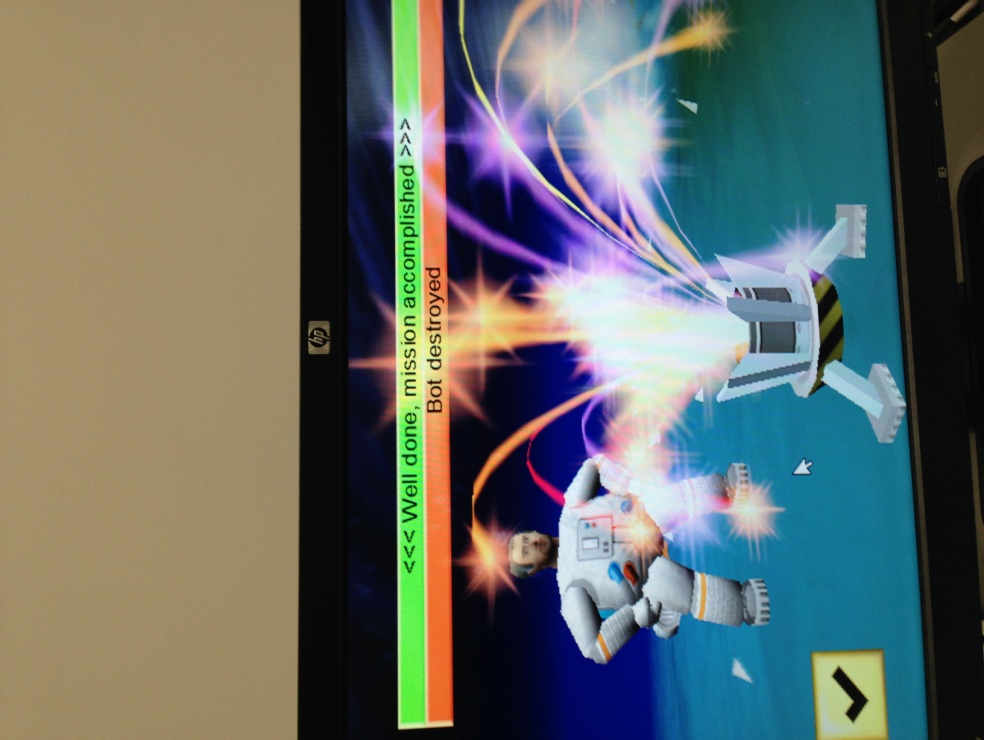
With regarding the way I wanted the grabber to turn I had to put a minus for the first angle as when you need to turn right in Ceebot by default it turns left so the use of a minus sign will make it turn in the opposite direction.

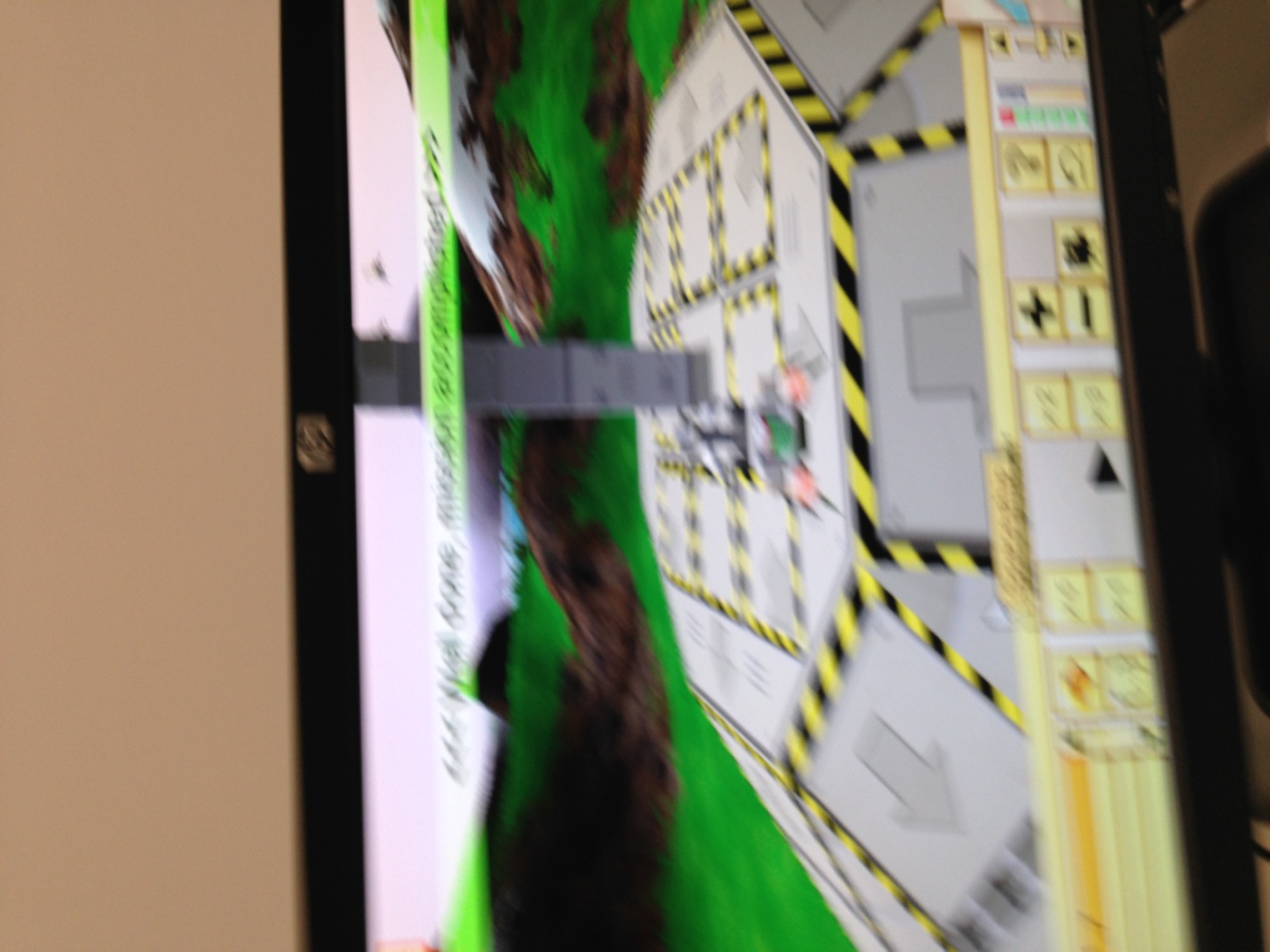




The grabber is on its way here to pick up the object ready to take back to the spaceship



Having got the tatiumn cell I walked back to the spaceship and get it to take off. However beware they are things like the poison mushrooms trying to kill you speed the robot up by pressing the x3 button on the Ceebot interface to make sure I did not die



I managed to get to the spaceship launch pad safely so I have successfully completed the mission

## Unit 1 Independent Study - Exercise 2 - Task 3.4 Production Chain

**Algorithm Logic**

Declare variables

grab the object

turn the first angle

move the first distance

drop the object

move the second distance

wait for a second

move the third distance

grab the object

move again

drop the object again

wait again for 15 seconds

grab the ready made object

move the distance requried

turn the right way round

move one last bit

drop the object

**Source Code**

On the next page is the source code for the independent study of unit 1 exercise 2 task 2 Production chain as I am having to do a lot of turning and moving I thought I would name the variables 1 2 and 3 as its easy to relate to in the program

extern void object::Task3\_4()

{

// Robert Collcott

// ID 21302939

// Course Computing

// Date 9th October

// Week 2 Exercise 3.4

float angle1;

float angle2;

float distance1;

float distance2;

float time1;

float time2;

angle1=90;

angle2=-90;

distance1 =3;

distance2 =-3;

time1 = 15;

time2 = 5;

grab();

turn(angle1);

move(distance1);

drop();

move(distance2);

wait(time1);

move(distance1);

grab();

move(distance2);

turn(angle2);

move(distance1);

drop();

wait(time2);

grab();

move(distance2);

turn(angle2);

move(distance1);

drop();

}



I have grabbed the titanium cube now I needed to covert this into a power cell by dropping it into the tatiumn mixer





After I made the power cell I had max energy to look and fire for trash and yes I found some and let the firing commence





Once I fired all the trash I completed the mission

## Unit 2 Input and Output - 3 Independent Study Tasks

## Unit 2 Independent Study - Task 1 - A modification of the draw rectangles program

**Source Code**

Here is the modified source code for the drawing rectangles program as you can see it now asks for user input for the width and length

extern void object::Task5\_5()

{

// Robert Collcott

// ID 21302939

// Computing

// Week 2 Exercise 5.5

// Date 21st ocober

float length=2;

float width=3;

float angle=90;

string input;

input=dialog("what length in meters would you like for the rectangle?");

length = strval(input);

input=dialog("what width in meters would you like for the rectangle?");

width = strval(input);

blue();

pendown();

move(length);

turn(angle);

move(width);

turn(angle);

move(length);

turn(angle);

move(width);

message("Rectangle of length"+ length+" meters width"+ width+ "of completed");

}



User input box should now appear when executing the program

I modified the original draw rectangles code to ask for some user input as before if you executed the program it would draw a rectangle without asking you for any particular length

## IMG_0578.JPG

As you can see now I have modified the program to ask for user input for the length and width of the rectangle in this case the rectangle is going to be 2 meters long and 3 meters wide

## 

## IMG_0579.JPG

The program asked me for what length and width for the rectangle

## 

## 

## IMG_0583.JPG

Lastly when I had finished putting in the length and width of the rectangle a user friendly message would come up. saying "Rectangle of length 2 meters and width of 3 has been completed

Completion message

## Unit 2 Independent Study - Task 2 Target Buster 2 – 5.2

**Source Code**

extern void object::Task5\_2()

{

// Robert Collcott

// ID 21302939

// Computing

// Week 2 Exercise 5.2

// 6th October 2014

object item;

float dist;

point tb;

point r;

item = radar (TargetBot);

tb = item.position;

item = radar (WheeledShooter);

r = this.position;

dist=distance(tb,r);

turn(direction(tb));

move(dist-15);

fire(1);

wait(0.5);

if (item!=null);

grab();

item = radar(Converter);

goto(item.position);

drop();

}

## Unit 2 Independent Study - Task 3 5.6 Target Destruction

extern void object::Task5\_6()

{

object item;

float dist;

point tb;

point r;

item = radar (TargetBot);

tb = item.position;

item = radar (LeggedShooter);

r = this.position;

dist=distance(tb,r);

turn(direction(tb));

move(dist-15);

fire(1);

wait(0.5);

}

## Unit 3 Independent Study - Tasks

In this unit of Ceebot there were 3 independent tasks I had to do on my own to practise the techniques more I had learned in the unit which in this case was Loops part 1. The three tasks I had to do were the following

* Shoot Out,
* Loop the Loop
* Shooting Practise

### 8.2 Shoot Out



I created a program that would move and fire targets 4 times automatically this way I will not have to keep executing it



Each target was destroyed after one another



When all the targets were destroyed mission was completed

## 8.6 Loop the Loop

This task I had to draw 3 squares inside each other by using the checkpoint markers in the world I had to program

**Source Code**

extern void object::Task8\_6()

{

// Robert collcott

// ID 21302939

// Computing

// Week 3 Exercise 8.6

// Date 22nd October 2014

int count;

int inc;

float size = 10;

float angle = 90;

pendown();

blue();

for (count=0; count<3;

count++)

{

for (inc=0; inc<4; inc++)

{

move(size);

turn(angle);

}

size = size + 5;

}

}

## IMG_0620.JPG

## 

I had to program the wheeled shooter to draw the squares

## IMG_0622.JPG

## 

One square drawn 2 more to draw had to put the count less than 4 as if it was three it would only have enough scope to draw 2 squares as the condition in the loop sated less than 3 and 2 is maths terms is less than 3

## IMG_0629.JPG

## IMG_0630.JPGUnit 3 Independent Study Task 3 - 11.3 Shooting Practise

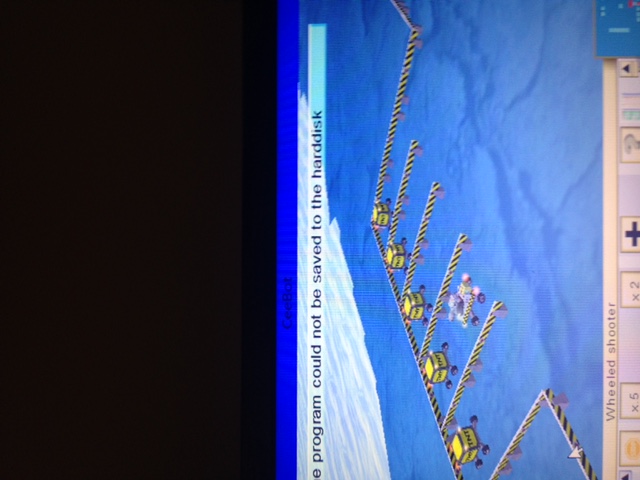
Once the all the squares have been drawn I completed the mission

Here you can the wheeled shooter has almost drawn 3 squares It has to programmed using variables so it knows how far too turn and draw the blue lines for the squares

Here you will see some sample images of the shooting practise program I had to do I had to destroy some targets by using the input from the user.



Here the shooter is going to shoot some targets



Shoot the targets down

## C:\Users\Robert Collcott\Desktop\ceebot11_3shootingpractiseexerciseevidence\IMG_1176.JPG

Carry on shooting until on the targets unitl mission completed

## Unit 4 Selection Independent Study Tasks

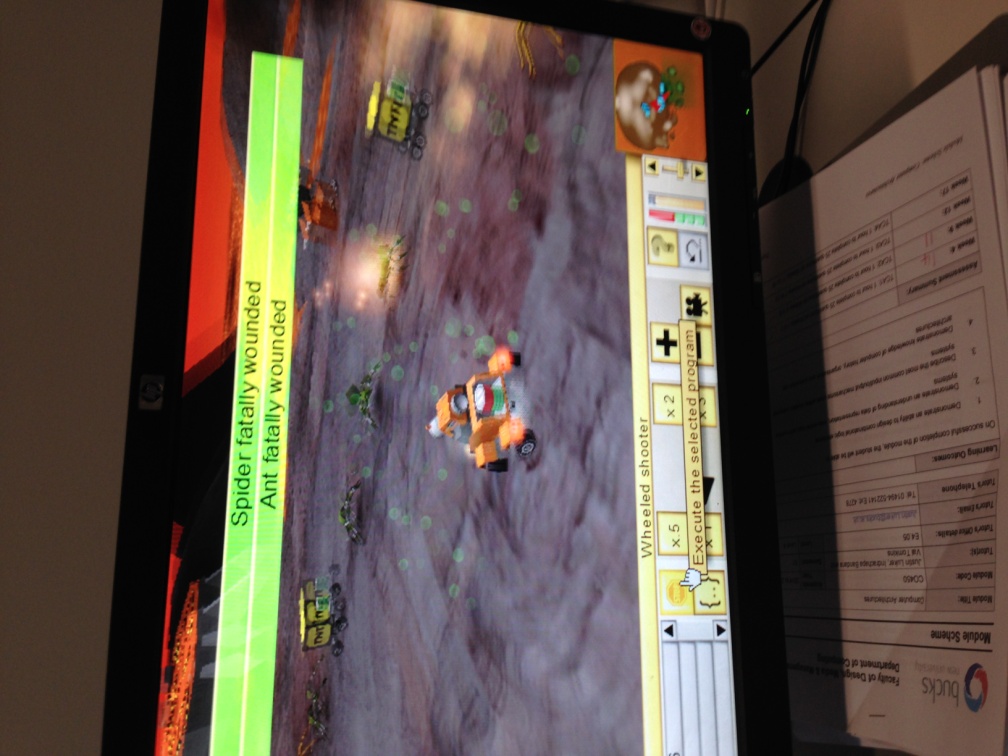
## 9.6 Alien Destruction

In this task I had to use a for loop as there were 20 objects and I wanted to repeat the following turn right fire and turn back to move onto the next object

They were 20 objects which were Target Bots and eggs and they were all positioned 5 meters apart so I programmed inside the for loop move 5 meters as any further would be too long and the shooter would fire but miss its target



I kept destorying the ants and spiders quickliy as if you do not shoot them with their fiece powers they can damage your bot's health level



Here you can see they were 20 objects but the ones I had to destroy were ants and spiders not the target bots



Once I destroyed all the objects the mission was completed and I can move on to the next task I have to do

7

## 5.5 A further modification of the draw variables rectangle program

I have already created a program that will draw rectangles of any length but the problem is if the length and width are too big the bot will hit the barriers and blow up

So I am going to do several tests to see what is the right length for the rectangle to be drawn

extern void object::Task5\_5()

{

// Robert Collcott

// ID 21302939

// Computing

// Week 4 Exercise 5.5

// Date 29th October

float length=2;

float width=3;

float angle=90;

string input;

input=dialog("what length in meters would you like for the rentangle?");

length = strval(input);

input=dialog("what width in meters would you like for the rentangle?");

width = strval(input);

if(length=!)==2meters) (width==3meters);

blue();

pendown();

move(length);

turn(angle);

move(width);

turn(angle);

move(length);

turn(angle);

move(width);

message("Rectangle of length"+ length+" meters width"+ width+ "of completed");

else(<=!rentanglemassive);

input=dialog("what length in meters would you like for the rentangle?");

length = strval(input);

input=dialog("what width in meters would you like for the rentangle?");

width = strval(input);

message ("The length is too big");

message ("The width is too massive");

}

# Unit 5 Loops part 2 Independent Study 3 Tasks

Based on my skills I acquired during this CeeBot unit I had to complete 3 extra independent study tasks

## 12.7 Ant Attack

extern void object::Task12\_7()

{

// Robert Collcott

// ID 21302939

// Computing

// Week 5 Exercise 12.7

// 2nd Decemeber 14

object item;

item=radar(WayPoint);

goto (item.position);

for(int count=0;count<10;)

{

item=radar(AlienAnt);

if (item!=null)

{

turn(direction(item.position));

fire(1);

wait(0.1);

}

}

}

## 15.1 Zig Zag

Algorithm Logic

Delcare variables

Turn 45 degreess background

Source Code

// Robert Collcott

// ID 21302939

// Computing

// Week 5 Exercisee 15.1

// 28th Jan 2015

int i=0;

object item;

turn(-45);

do

{

item=radar(GoalArea);

distance(this.position,item.position);

while(i<8)

{

if(i==0||i==2||i==4||i==6||i==8)

{

move(7);

turn(90);

}

if(i==1||i==3||i==5||i==7)

{

move(7);

turn(-90);

}

i++;

}

move(7);

}

while(item==null);

}

## 14.7 Autopilot

**Source Code**

// Robert Collcott

// ID 21302939

// Computing

// Week 5 Exercisee 15.1

// 28th Jan 2015

object item;

point airpt,botpos,spaceship;

float distance,direction;

item=radar(AirPoint);

while(item!=null)

{

while(this.position.z<item.position.z)

{

jet(0.4);

}

while(this.position.z>item.position.z)

{

jet(-0.4);

}

jet(0);

botpos=this.position;

airpt=item.position;

distance=distance(botpos,airpt);

turn(direction(airpt));

move(distance+5);

item=radar(AirPoint);

}

item=radar(SpaceShip);

spaceship=item.position;

distance=distance(botpos,spaceship);

goto(spaceship);

while(this.position.z>item.position.z)

{

jet(-0.4);

}

jet(0);

}

Sample screenshots



## 



## Unit 6 Functions - 2 Independent Study Tasks

Here are my independent studies for the functions module of Ceebot functions can be used to break up big programs into smaller more manageable units

This means they can be used over and over again in different programs. They can also be used to return the same result back to the user

Functions are particularly useful to break up larger programs

## 19.3 One More Flower

}// Robert Collcott

// ID 21302939

// Computing

// Week 6 Exercise 19.3

// 6th November 2014

Source Code

{

int i;

move(12);

turn(90);

move(0.5);

for(i=0;i<18;i++)

{

PetalA();

PetalB();

}

}

void object::PetalA()

{

int i=0;

yellow();

pendown();

while(i<53)

{

move(0.1);

turn(-10);

i++;

}

turn(180);

}

void object::PetalB()

{

int i=0;

red();

pendown();

while (i<53)

{

move(0.1);

turn(-10);

i++;

}

turn(180);

}





## 19.4 Triangle function

Here we are going to make a Maltese cross by creating a triangle function

I had to put to functions in the program these are

Move to Center()

This function should move the robot to the center of the floor space

Triangle()

This function I designed to draw one equilateral triangle with 5 meter long sides

**Source Code**

// Robert Collcott

// ID 21302939

// Computing

// Week 6 Exercise 19.4

// 6th November 2014

extern void object::Task19\_4()

{

int count;

MoveToCentre();

for(count=0;count<6;count++)

{

Triangle();

}

}

void object::MoveToCentre()

{

move(12);

turn(90);

move(7);

turn(-90);

}

void object::Triangle()

{

int count;

black();

pendown()

;

for(count=0;count<3;count++)

{

move(5);

turn(120);

}

turn(-90);

}

Sample screen shots of One More Flower

## 19.4 Triangle function extra

I am going to change the shape from an equlatiual triangle into a hexagon this time I need to change the colour of the shape from black to red

{

int count;

MoveToCentre();

for(count=0;count<6;count++)

{

Triangle();

}

}

void object::MoveToCentre()

{

move(12);

turn(90);

move(7);

turn(-90);

}

void object::Triangle()

{

int count;

red();

pendown()

;

for(count=0;count<3;count++)

{

move(5);

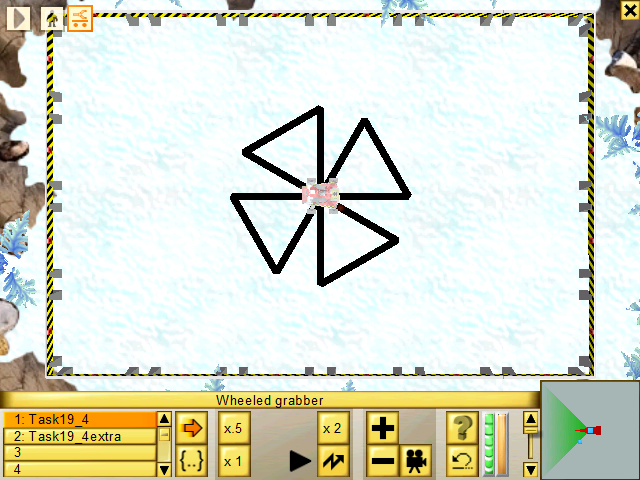
turn(120);

}

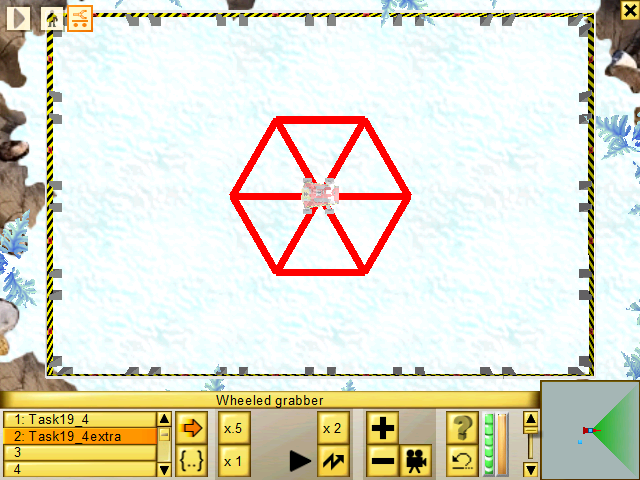
turn(-90);

}

Here is the trinabgle drawn using functions to make the program more flexiable



Here is the trinagle I used to draw using separate

19.4 Extra

Here is the hexagon I drew using functions

## 

{

int count;

MoveToCentre();

for(count=0;count<6;count++)

{

Hexagon();

}

}

void object::MoveToCentre()

{

move(12);

turn(90);

move(7);

turn(-90);

}

void object::Hexagon()

{

int count;

red();

pendown()

;

for(count=0;count<6;count++)

{

move(5);

turn(120);

}

turn(-60);

}

## Unit 7 Functions and Parameters - Independent Study Tasks

## 21.6 The Fly to Function

This exercise there is a **Winged Grabber** robot that has to fetch various items from nearby islands

The robot needs to do the following

* Drop the Titanium at the red flag
* Drop the Titanium Ore at the yellow flag
* Drop the Fuel Cell at the blue flag
* Drop the power cell at the green flag

**Algorthim**

Delcare variables

Define functions

Fly to the red flag to dropn the titanium at the red flag

Fly to the yellow flag to drop TitaniumOre

Fly to the blue to drop the blue cell

Fly to the green flag to drop the power cell off

Use a robot’s radar to dectect the object area

Display a message saying I am off to the certain object

**Extra**

Add to function

Grab and drop

**Source code**

extern void object::Task21\_6()

{

// Robert Collcott

// ID:21302939

//Week 7 Exercise 21.6:

//Date 13/nov/14:

int tt1,f1,to2,f2,fc3,f3,pc4,f4;

tt1=Titanium;

f1=RedFlag;

to2=TitaniumOre;

f2=YellowFlag;

fc3=FuelCell;

f3=BlueFlag;

pc4=PowerCell;

f4=GreenFlag;

FlyTo(tt1);

FlyTo(f1);

FlyTo(to2);

FlyTo(f2);

FlyTo(fc3);

FlyTo(f3);

FlyTo(pc4);

FlyTo(f4);

}

void object::FlyTo(int selection)

{

object item=radar(selection);

message("On my way to "+item.category);

goto(item.position);

wait(0.5);

if(item.position.y>0)

{

grab();

}

else

{

drop();

}

wait(0.5);

}

## 21.6 The Fly to Function extra

NOT ATTEMPTED

# 

# 

The winged grabber is here on it its way to the titanium

# 

Having collected the titanium it is now going to drop it at the red flag



The winged grabber is on its way now to the next point to drop off the next product which is Titianum Ore

# Unit 8 - Arrays – Independent Study tasks

Here are my independent study tasks for unit 8 of Ceebot

## Unit 8 Arrays exercises conversion to use functions

In this Ceebot module we have to convert all of the exercises to use functions to make them more efficent to use and matian

HAD BIG TROUBLE DOING THIS ONE – HAD A GO BUT COULD NOT GET IT TO WORK

# Unit 9 Ceebot Project ---

This was my final part of my Ceebot introduction to the programming concepts module I had to program a robot that would follow an enemy to get the staff biscuits back.

In order to do this I had to spilt the program into 4 chunks one for the basic control system which is the basic program that I had to create so the user had control to catch the enemy bot

**Basic program algorithm**

Declare variables

define functions

get a key press to go left right up or down

move the distance required

follow the enemy until he drops the box

**Basic Source Code**

extern void object::Chase()

{

// Robert Collcott

// ID 21302939

// Computing

// Week 10 Exercise 24.2

// Ceebot Project

// Thursday 4th December 2014

UserInstructions();

}

void object::UserInstructions()

{

while(true)

jet(0);

message("Up to fly up");

message("Down to fly up");

message("Left to fly left");

message("Right to fly right");

if(keypushed(VK\_UP)) // Fly up to start following the enemmy

{

jet(1);

wait(0.01);

}

{

if(keypushed(VK\_LEFT))

// Turn left to try and catch the enemey

{

turn(4);

wait(0.02);

}

}

{

if(keypushed(VK\_RIGHT))

// Turn right to try and catch the enemey

{

turn(-4);

wait(0.02);

}

}

{

if(keypushed(VK\_DOWN))

// Come down to land to get the box

{

jet(-0.1);

jet(0);

wait(0.02);

}

}

}

void object::ComeintoLand()

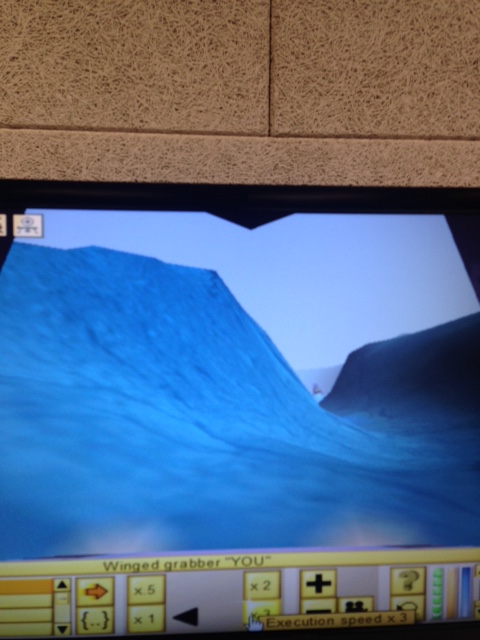
{

jet(0.1);

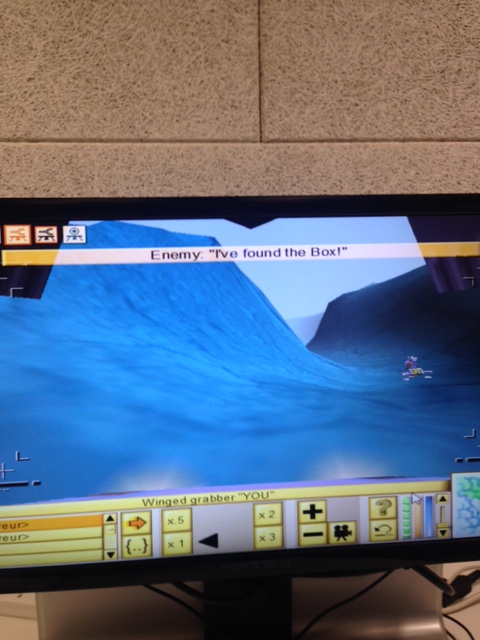
grab();

}

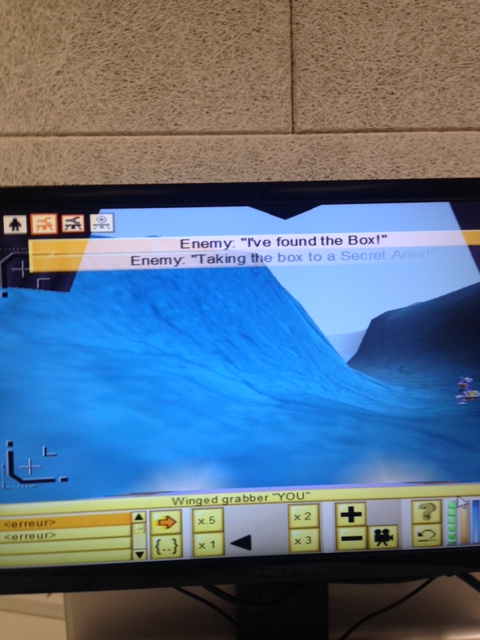
**Project pictures of the basic program**



Here you can see I am starting on my hunt to find the enemy



The enemy is off – no I need to catch him before he steels the biscuits!!!!!



The enemy is off – no I need to catch him before he steels the biscuits!!!!!



Here the user can even choose to fly manualy or have the computet do it for them

A = Auto

M = Manual – **User does it themselves**

# Project Extension Work -- Part 1

I need to here find a way of slowing down before grabbing the black box which contains the Buckinghamshire New University computing department's biscuits.

So I must provide a way of preventing a way of my robot

**Source Code**

message("Press key G to grab");

if (keypushed(VK\_G))

{

grab(BlackBox);

}

message("Press key Z to drop the box");

if (keypushed(VK\_Z))

{

drop (BlackBox);

}

Additional functinolity was put In here for Ease of use to grab and drop the black box

# Project Extension Work -- Part 2 – Autopilot

**Source Code**

string input;

string A;

string M;

message("If you select auto pilot please wait");

input=dialog("Please press M for manual or A for Auto");

input=strupper(input);

if(input=="A")

{

Autopilot();

}

if(input=="M")

{

Manual();

}

void object::Autopilot() //Autopilot function

{

Takeoff();

Follow();

ReturnHome();

}

//\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

void object::Takeoff ()

{

float height=25;

jet(0.5);

while(this.altitude<height);

{

wait (0.1);

}

jet(0);

}

//\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

void object::Follow ()

{

object item;

float dist;

while(true)

{

item=radar(WingedGrabber);

turn(direction(item.position));

(distance(position,item.position));

if(distance(position,item.position)>=30)

{

drive (1,0);

}

else

if(distance(position,item.position)<=30)

{

drive (0,0);

wait (0.1);

}

wait(1);

item=radar(BlackBox);

if(item!=null)

{

break;

}

}

move(distance(position,item.position));

}

void object::ReturnHome ()

{

object item;

point itemposition;

float height=25;

jet(-0.5);

while(this.altitude<height)

{

wait(0.1);

item=radar(BlackBox);

itemposition=item.position;

turn(direction(itemposition));

goto(itemposition);

wait(0.1);

grab();

{

break;

}

}

object SpaceShipReturn;

point SpaceShipLocation;

SpaceShipReturn=radar(SpaceShip);

SpaceShipLocation=SpaceShipReturn.position;

goto(SpaceShipLocation);

drop();

}

I continued to follow the enemy until he dropped the staff biscuits



Now I got the enemy tracked down I can pick up the black box and take it to the spaceship

# 

# 

# Final piece of project - extension work 3 - finish off the enemy

**Algorithm**

Create function destroyEnemy

Define object variable

Detect enmey’s radar

Start a loop

While the enemy postion is less than item position

Ascend very slowly

Slow down compeltly

Call Follow the enemy function

Detect object item variable

Define distance variable

Start another loop

Find the enemy with a a radar

Turn in the direction of the enemy

Travel the distance of the enemy

Distance greater than 15

Move back 15 meters

Break

Call destroy grabber function

Declare object variable

Find the enemy with a radar

Turn in the direction of the enemy

Pause for 1 minute

**Destroy the enemy**

**Source Code**

{  
destroyEnemy();  
}  
  
void object::destroyEnemy()   
{  
TakeOff();   
Follow();  
DestroyGrabber();  
}  
  
void object::TakeOff()  
{  
object item;   
  
item=radar("Enemy");  
while(this.position.z<item.position.z)   
{  
jet(0.1);  
}  
jet(0);   
}  
  
void object::Follow()   
{  
object item;  
float distance;  
  
while(true)   
{  
item=radar("Enemy");  
turn(direction(item.position));  
distance=distance(this.position,item.position);  
  
if(distance>15)  
{  
move(distance-15);  
break;  
}  
}  
}  
  
void object::DestroyGrabber()  
{  
object item;   
  
item=radar("Enemy");   
turn(direction(item.position));   
wait(1);  
fire(1);  
}

Get to the enemy here

Destroy the enemy here

Here is the Winged Shooter this is what killed the enemy which stole the staff biscuits

**C# Console**

## Unit 1 C# Console – Input and Output

This unit is the introduction to the C# programming language with input and output as well there is two independent study tasks for me to complete and document

## 1.6 - Rocket Game - Independent Study

**Algorithm Logic**

Declare 4 variables

* Length
* width
* height
* price

Input length of tank

Input width of tank

Input height of tank

Calculate volume of tank

Display and calculate the total cost of fuel tank

calculate and display the time in seconds that the fuel is going to lasts

Finally output the details on the console screen

**Source Code**

Here is my source code for the rocket game program

namespace \_1.\_6\_Rocket\_Game

{

class Program

{

// Robert Collcott

// ID 21302939

// Computing

// Date 11/11/2014

static void Main()

{

string input;

double length, width, height, price;

double volume, time;

const int rate = 6;

Console.Write("The length of the fuel tank in metres: ");

input = Console.ReadLine();

length = Convert.ToDouble(input);

Console.Write("The width of the fuel tank in metres: ");

input = Console.ReadLine();

width = Convert.ToDouble(input);

Console.Write("The height of the fuel tank in metres : ");

input = Console.ReadLine();

height = Convert.ToDouble(input);

Console.Write("The price of the rocket fuel tank : ");

input = Console.ReadLine();

price = Convert.ToDouble(input);

Console.Write("The volume of the fuel needed to fill the tank is" + length \* width \* height);

volume = length \* width \* height;

Console.Write("\nThe total cost to fill the tank is £" + price \* volume);

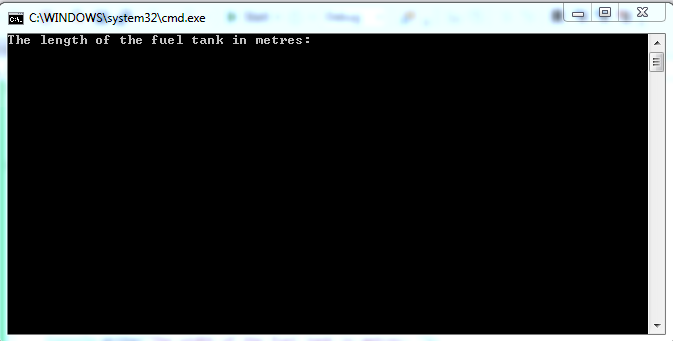
Console.Write("\nThe rocket fuel will last for " + volume / rate + " seconds.");

Console.WriteLine();

}

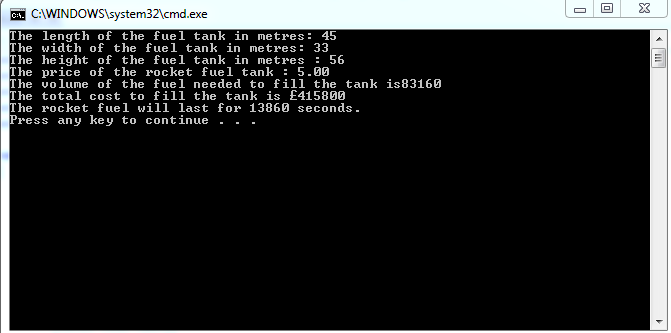
}

}



Here you can see the user will have to input the length of the fuel tank the height the width the price all have to be inputted to calculate the total weight

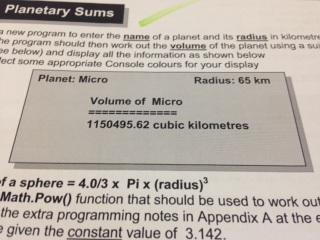
The way this will happen is the user will type in the values in



Here you can see with all the input done the program calculates it works out the volume and how long the fuel tank will last for

## 1.7 Planetary Sums - Independent Study

**Sample screen output of Planetary sums**



**Algorithm Logic**

Declare variables

Get user input to enter the name of the planet

Get user input to input the planet radius in kilometers

Display and output the volume of the planet

Output the final result

**Source Code**

namespace \_1.\_7\_Planetary\_Sums\_new

{

class Program

{

// Robert Collcott

// ID 21302939

// Computing

// Week 1 exercise 1.7

// 15th January 2015

static void Main()

{

string input, name;

double radius, volume;

const double Pi = 3.142;

Console.Write("Enter the name of a planet:");

name = Console.ReadLine();

Console.Write("Enter the planet radius in kilometres: ");

input = Console.ReadLine();

radius = Convert.ToDouble(input);

volume = 4.0 / 3 \* Pi \* (radius \* radius \* radius);

Console.WriteLine();

Console.WriteLine("\tPlanet: " + name + "\t\t\t\t Radius: " + radius);

Console.WriteLine("\n\t\t\tVolume of " + name);

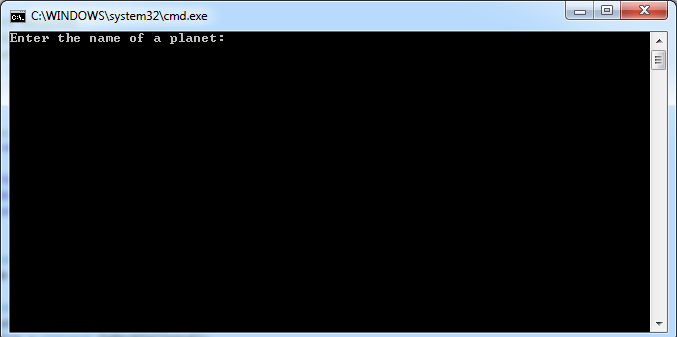
Console.WriteLine("\n\t\t\t========== \n\n\t\t\t" + volume.ToString("0.00") + " cubic kilometres");

Console.WriteLine();

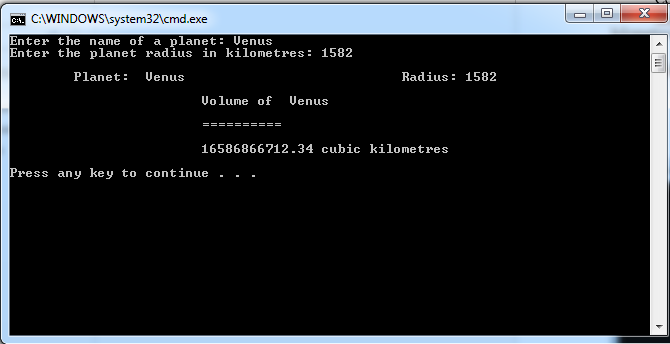
}

}

}



Here the user is going to have an enter the name of the planet e.g. Satuan or Mars



Once I have entered the planet and entered the planets radius it is going to work out what the volume surface area of the planet is kilometers

it will out this work by timesing the two together in volume cubic meters

class Program

{

// Robert Collcott

// ID 21302939

// Computing

// Week 1 exercise 1.7

// 15th January 2015

static void Main()

{

string input, name;

double radius, volume;

const double Pi = 3.142;

Console.Write("Enter the name of a planet:");

name = Console.ReadLine();

Console.Write("Enter the planet radius in kilometres: ");

input = Console.ReadLine();

radius = Convert.ToDouble(input);

volume = 4.0 / 3 \* Pi \* (radius \* radius \* radius);

Console.WriteLine();

Console.WriteLine("\tPlanet: " + name + "\t\t\t\t Radius: " + radius);

Console.WriteLine("\n\t\t\tVolume of " + name);

Console.WriteLine("\n\t\t\t========== \n\n\t\t\t" + volume.ToString("0.00") + " cubic kilometres");

Console.WriteLine();

}

}

}

## Unit 2 C# Console - Designing programs

This unit I learnt about designing programs in the C Sharp programming language the three phases I learnt about which were

* Designing the Program
* The program being coded
* Testing the program

The program design phase was all about seeing what will the necessary input and outputs for the program the variables the program was going to have

Lastly a algorithm in structured English so the developers could see what the problem was and how to tackle it

The next step was to actually code the program by using the design document produced and lastly and the final step was to test the program to see if the right inputs produced the right outputs of results

## 2.4 Food for thought independent study task Unit 2 C# Console

Program Name

Food for thought:

Program objective

To calculate the total cost of meals by asking the user to enter the name of the restaurant and the separate costs for the starter main course and sweet **[desert]**

Input-Output Diagramshowing the inputs and outputs for the program

Starter

Tax

Program

Main Course

Sweet

Total

Cost

Identifier Listthe variables to be used in the program

|  |  |  |
| --- | --- | --- |
| Identifier | **Data Type** | **Meaning** |
| **Cost** | String | The cost of each part of what the people ordered |
| **Starter** | Double | The starter that people ordered |
| **Main Course** | Double | The main that people ordered |
| **Sweet** | Double | The sweet that people ordered |
| **Total** | Double | The total cost of all the choices people ordered |
| **Tax** | Double | The 12% tax added in with the bill |
| **Grand Total** | Double | This is the total with the Tax added in |

Algorithm

Stepwise method for solving the problem in structured English (or pseudocode)

Get user input for users name

Get user input for resturant name

Get user input for starter

Get user input for main

Get user input for sweet

Print out the total without the VAT

Ask for input for the tax

Calculate all the values include tax

Display the total cost of everything

Output the final result with the tax included

Test Planactual results are left blank at this stage and filled in after the program has been written.

Finally you can test your coded program and fill in the results obtained in the Test plan

|  |  |  |  |
| --- | --- | --- | --- |
| **Starter** | | | |
| Test No. | **Inputs** | **Expected Outputs** | **Actual Outputs** |
| 1 | 4.50 |  |  |
| 2 | 8.99 |  |  |
| 3 | 5.00 |  |  |
| 4 | 18.49 |  |  |
| 5 |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Main Course** | | | |
| Test No. | **Inputs** | **Expected Outputs** | **Actual Outputs** |
| 1 |  |  |  |
| 2 |  |  |  |
| 3 |  |  |  |
| 4 |  |  |  |
| 5 |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Sweet** | | | |
| Test No. | **Inputs** | **Expected Outputs** | **Actual Outputs** |
| 1 |  |  |  |
| 2 |  |  |  |
| 3 |  |  |  |
| 4 |  |  |  |
| 5 |  |  |  |

## Source Code

static void Main(string[] args)

{

string input, name;

double starter, main, sweet, total, tax, grand;

Console.Write("The name of the resturant today is:");

name = Console.ReadLine();

Console.Write("The cost of the starter course today is:");

input = Console.ReadLine();

starter = Convert.ToDouble(input);

Console.Write("The cost of the mainb course today is:");

input = Console.ReadLine();

main = Convert.ToDouble(input);

Console.Write("The cost of the sweet course today is:");

input = Console.ReadLine();

sweet = Convert.ToDouble(input);

total = (starter + main + sweet);

tax = (total/ 100 \* 12);

grand = (total + tax);

Console.Write("The cost of the meal today is:" + total);

}

}

}

## Unit 3 C# Console Sequence Selection and Iteration

In this module of C# Console I learnt about the ways that the C# language uses Sequence Selection and Iteration statements so programs can be working for longer periods of time

## Unit 3.6 Game On

**Algorithm**

Get user input the name of the players name

Computer picks a random number

Get user input for the player’s guess

Compare players guess with computer random generated number

Message appears saying too high too low or SPOT ON

Start a loop

Repeat steps until the players guess is correct

Tell the player how many guesses the player took

Ask for input if a user wants to have a go

Start another loop

Repeat the whole program while the input is not n for no

Output the number of players who have guessed the game at the end

**Source code**

static void Main()

{

string input, name, again = "Y";

int guesses, guess, number, plays = 0;

Random ranNum = new Random();

while (again == "Y")

{

plays++;

Console.Write("What is your name? ");

name = Console.ReadLine(); number

= ranNum.Next(100) + 1; guesses = 0;

Console.Write(name + "\n I am going to pick a number between 1 and 100");

Console.Write("\nYou must try to guess the number");

Console.Write("\nOK.. I have picked a number");

do

{

guesses++;

Console.Write("\nGuess the number? ");

input = Console.ReadLine();

guess = Convert.ToInt32(input);

if (guess > number)

{ Console.WriteLine("TOO HIGH " + name); }

else if (guess < number)

{ Console.WriteLine("TOO LOW " + name); }

else

{

Console.WriteLine("SPOT ON " + name);

if (guesses >= 10)

{

Console.WriteLine("You took " + guesses + " guesses. That's terrible!!");

Console.ReadLine();

}

else if (guesses < 5)

{

Console.WriteLine("You took " + guesses + " guesses. Very good!!");

Console.ReadLine();

}

}

}

while (guess != number);

Console.WriteLine("Would you, or anyone else like to play again? (Y/N)");

again = Console.ReadLine(); again.ToUpper();

}

Console.WriteLine("========================");

Console.WriteLine("Thanks for playing");

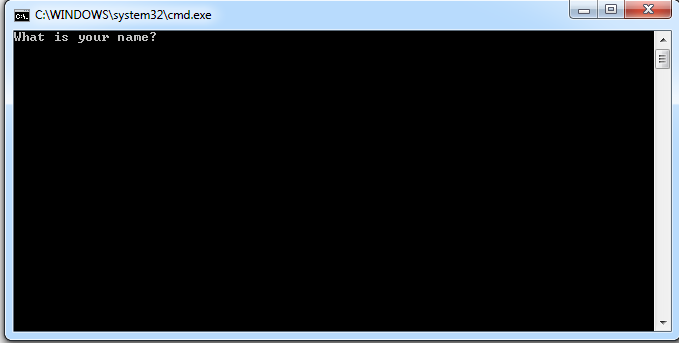
Console.WriteLine("You played " + plays + " times");

Console.ReadLine();

}

}

}



The program starts off with you entering your name to play the game



The program has a random class in called randy this picks a number between 1-100

## 

If a player got to low a message would say to low if it was too high a message would say too high or if the certain user who was playing got it spot on

A message would be displayed saying WELL DONE YOU GOT IT SPOT ON

## 

## Unit 3.7 Game choices

**Algorithm**

Get user input for the player’s name

Display a menu of 4 options

1. **Advanced**
2. **Experienced**
3. **Average**
4. **Novice**

Ask for user input of chosen skill level by choosing options 1 to 4

User inputs choice of level 1 to 4

If user inputs 1 - 4

Display thank you message with the chosen level and ask is the level you want

else

Display a message saying sorry **user**  this game only has levels 1 - 4

Display message if the user

When the answer is yes the player with name eg [xxx.xxx] can start the game

**Source code**

static void Main(string[] args)

{

int level;

string input, name;

Console.Write("Please enter player name");

name = Console.ReadLine();

Console.Write("Please enter level to play");

input = Console.ReadLine();

level = Convert.ToInt32(input);

if (level < 5)

{

Console.Write("Thank you you have chosen level " + level + " in the game");

Console.Write("Is the level you want?");

input = Console.ReadLine();

}

else

Console.WriteLine("Sorry they are 4 levels in this game only");

}

}

}

## Unit 4 C# Console Classes Objects and Methods

This is the last bit of the C# Console independent studies

## 4.6 Nuclear Control

**Algorithm**

Define variables

Create classes and objects

Define menu

Display menu options

1. Lower fuel rods

2. Raise Fuel rods

3. Activate Shields

4. Deactivate Shields

5. Quit the program

Start a loop

Choose option 1-5 return entry back

Validate user input

If user hits 1 as the choice call the lowerRods method

Put the correct code in otherwise you are in the shit

HAD A GO AT BOTH OF THESE BUT COULD NOT GET THE PROGRAMS TO WORK

## 4.7 More Craps!!

**Algorithm Logic**

Declare variables

Declare Classes and Objects

Test

Station

Call my station method

Display console screen with the following options

1. Lower Fuel Rods
2. Raise Fuel rods
3. Activate shields
4. Deactivate shields
5. Quit

Call my station’s **getChoice()** method

Ask for user input to enter a choice of 1-5

Return entry to the user

If

A user enters choice 1 the lowerRods method will be called the user has to enter the correct code and if you do not get it correct you are in trouble

Else

Input is correct continue on with the same option or pick a different one

## 4.7 More Craps

**Algorithm Logic**

Define CrapsGame class

Create a constructor method

Create random object to place in the main

Create 2 methods

Remove the main method from craps

Crerate method in manyPlay()

Create updateScore() method in Craps game

Output games won and lost

Return value from the play method

Display the