

# CO452 CeeBot Classwork

## Task 4\_1: Follow Path

### Program Description

Program the **Wheeled Grabber** to reach the platform that you can see in the distance. The path to take is marked out with blue waypoints (checkpoints). These are all 20 metres apart

### Program Algorithm

1. Set angle to 90
2. Set distance to 20
3. Move forward by distance units
4. Turn angle by degrees
5. Move forward by distance units
6. Turn angle by -degrees
7. Move forward by distance units
8. Turn angle by -degrees
9. Move forward by distance units

### Program Code

```
6-5 Target Practice.cs x 3-4 Production Chain.cs x 4-3 Using the Radar.cs x 5-5 Drawin
1 // Derek Peacock ID 123456
2 // FD Computing 2018
3 // CO452 Programming Concepts
4 // Week 1 Task 4.1 Follow Path
5 // 09/10/2018
6
7 int length = 20;
8 int angle = 90;
9
10 move(length);
11 turn(angle);
12
13 move(length);
14 turn(-angle);
15
16 move(length);
17 turn(-angle);
18
19 move(length);
20
```

## Task 1\_4: Move an Object

## Program Algorithm

**Algorithm**

- 1. Store 2 in variable a**
- 2. Store 8 in variable b**
- 3. Subtract 2 from variable b**
- 4. Multiply a by b and store result**
- 5. Display the result**

## Program Code

```
1 // Derek Peacock ID 123456
2 // FD Computing 2018
3 // CO452 Programming Concepts
4 // Week 1 Task 5.3 Follow Path
5 // 09/10/2018
6
7 int a = 2;
8 int b = 8;
9
10 int result;
11
12 b = b - 2;
13 result = a * b;
14
15 message("Results = " + result);
16
```

## Task 6.3 Step by Step

### Program Description

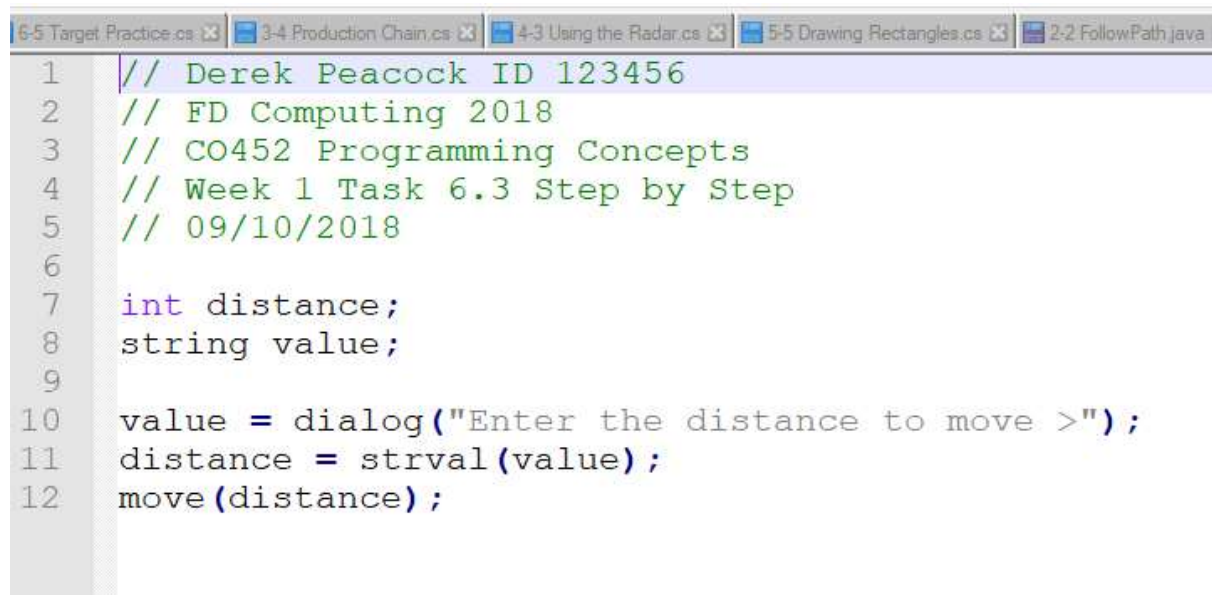
This is another example of moving, but this time you have to use the dialog box to input numbers and also convert them from a string type to a number type. Your WheeledGrabber is at one end of the track and there are mines at the other end, just past a platform .. your destination.

Your task is to reach the platform and survive, by asking the user to input a distance to move, then to move the robot through that distance. If you keep running the program, you should eventually reach your destination.

### Algorithm

1. Ask the user to input the distance to move
2. Move forward that distance

### Program Code



```
1 // Derek Peacock ID 123456
2 // FD Computing 2018
3 // CO452 Programming Concepts
4 // Week 1 Task 6.3 Step by Step
5 // 09/10/2018
6
7 int distance;
8 string value;
9
10 value = dialog("Enter the distance to move >");
11 distance = strval(value);
12 move(distance);
```

## Task 1.6 Power up a Robot

### Program Description

Use the **Wheeled Grabber** to pick up a power cell and drop it on the **Winged Grabber**

### Algorithm

1. Move a distance of 7
2. Grab the power cell
3. Turn 90 degrees
4. Drop the power cell

### Program Code

```
6-5 Target.Practice.cs x 3-4 Production Chain.cs 4-3 Using the Radar.cs 5-5 Drawing Rectangles
1 // Derek Peacock ID 123456
2 // FD Computing 2018
3 // CO452 Programming Concepts
4 // Week 1 Task 1.6 Power up a Robot
5 // 09/10/2018
6
7 move (7) ;
8 grab () ;
9 turn (90) ;
10 drop () ;
11
12
```

## Task 6.4 Think of a Number

### Program Algorithm

#### **Algorithm**

1. Input any value from the user and convert this to a float number
2. Double this number and store the result separately
3. Add 16 to the previous result
4. Divide the result by 2
5. Subtract the original number from this result
6. Display the final answer

### Program Code

```
5 Target Practice.cs x 3-4 Production Chain.cs x 4-3 Using the Radar.cs x 5-5 Drawing Rectangles.cs x 2-2 FollowPath.java x
1 // Derek Peacock ID 123456
2 // FD Computing 2018
3 // CO452 Programming Concepts
4 // Week 1 Standard Task 6.4 Power up a Robot
5 // 09/10/2018
6
7 string value;
8 int originalNumber;
9 int currentNumber;
10
11 value = dialog("Enter a number > ");
12 originalNumber = strval(value);
13
14 currentNumber = originalNumber * 2;
15 currentNumber = currentNumber + 16;
16 currentNumber = currentNumber / 2;
17 currentNumber = currentNumber - originalNumber;
18
19 message("Answer = " + currentNumber);
20
```

## Task 6.5 Target Practice

### Program Algorithm

#### Algorithm

1. Input the vertical angle (-20 to 20)
2. convert this to a float number
3. Input the horizontal angle (-90 to 90)
4. Convert this to a float number
5. Aim the shooter
6. turn the shooter
7. Fire
8. Set the aim back to horizontal
9. Turn back to the starting position
10. Output a message "Ready to fire again"

### Program Code

```
6-5 Target Practice.cs | 3-4 Production Chain.cs | 4-3 Using the Radar.cs | 5-5 Drawing Rectangles.cs | 2-2 FollowPath.java | 5-3 Robot M
1 // Derek Peacock ID 123456
2 // FD Computing 2018
3 // CO452 Programming Concepts
4 // Week 1 Standard Task 6.5 Target Practice
5 // 09/10/2018
6
7 string value;
8 float vertical;
9 float horizontal;
10
11 value = dialog("Enter Vertical Angle (-20..20) >");
12 vertical = strval(value);
13
14 value = dialog("Enter Horizontal Angle (-90..90) >");
15 horizontal = strval(value);
16
17 aim(vertical);
18 turn(horizontal);
19 fire();
20
21 aim(0);
22 turn(0);
23
24 message("Ready to fire again!");
25
```

## Task 4.3 Using the Radar

### Program Algorithm

#### Programming Concepts CO452

##### Algorithm

1. Use radar to find position of a TitaniumOre object
2. Go to this position
3. Pick up the object
4. Use radar to find position of the Converter
5. Go to this position
6. Drop Titanium onto Converter
7. Step back to allow converter to do its job

### Program Code

```
1-3 Using the Radar.cs x 5-5 Drawing Rectangles.cs x 2-2 FollowPath.java x 5-3 Robot Maths.java x 6-3 S
1 // Derek Peacock ID 123456
2 // FD Computing 2018
3 // CO452 Programming Concepts
4 // Week 1 Standard Task 4.3 Using a Radar
5 // 09/10/2018
6
7 object item;
8
9 // Find ore and get it
10
11 item = radar(TitaniumOre);
12 goto (item.position);
13 grab();
14
15 // move ore to convertor
16
17 item = radar(Converter);
18 goto (item.position);
19 drop();
20 move (-3);
21
```

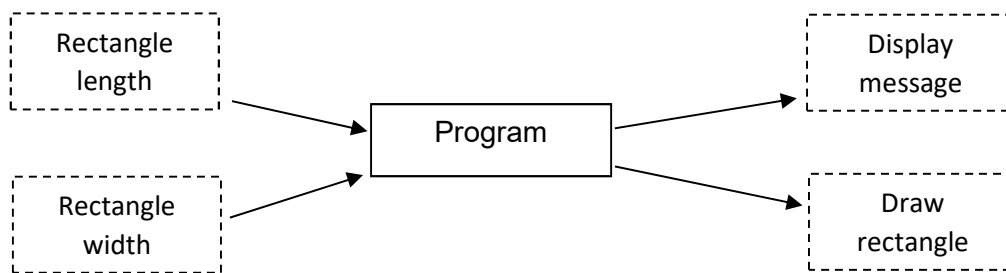
## Task 5.5 Drawing Rectangles

### Program Algorithm

1. Get the length of the rectangle
2. Convert it to a number
3. Get the width of the rectangle
4. Convert it to a number
5. Change to the blue pen
6. Put the pen down
7. Move the given length
8. Turn 90 degrees
9. Move the given width
10. Turn 90 degrees
11. Move the given length
12. Turn 90 degrees
13. Move the given width
14. Print a message

### Input-Output Diagram

showing the user inputs and outputs for the program



### List of Variables

Identifier	Type	Meaning
value	string	To get any value from the user
length	int	Length of the rectangle
width	int	Width of the rectangle
angle	int	The angle of one corner of the rectangle

### Test Plan

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

Test No.	Inputs		Expected Outputs		Actual Outputs	
1	Length = 20	Width = 15	Rectangle 20 x 15	Message	Rectangle 20 x 15	Message
2	Length = 10	Width = 8	Rectangle 10 x 8	Message	Rectangle 10 x 8	Message



## Program Code

```
4:3 Using the Radar.cs 5:5 Drawing Rectangles.cs 2:2 FollowPath.java 5:3 Robot Maths.java 5:3 Step by Step.java 1:6 Power Up Robot.java 5:4 ThinkNumber.cs
1 // Derek Peacock ID 123456
2 // FD Computing 2018
3 // CO452 Programming Concepts
4 // Week 1 Standard Task 5.5 Draw Rectangles
5 // 10/10/2018
6
7 int length;
8 int width;
9 string value;
10
11 value = dialog("Enter rectangle length >");
12 length = strval(value);
13
14 value = dialog("Enter rectangle width >");
15 width = strval(value);
16
17 blue();
18 pendown();
19
20 move(length);
21 turn(90);
22
23 move(width);
24 turn(90);
25
26 move(length);
27 turn(90);
28
29 move(width);
30
31 message("Rectangle length " + length + " metres and width " + width + "metres completed");
32
33
```