Task 1: Modelling (25 marks)

Task 1.1: Cells (These are cells in the warehouse in the format cix where i is row and x is column)

c51
c52
c53
c54
c55
c41
c42
c43
c44
c45
c31
c32
c33
c34
c35
c21
c22
c23
c24
c25
c11
c12
c13
c14
c15
scanner -a scanner
p001 - a package
p010 - a package
p011 - a package
belt - the delivery belt
switch - switch
bot - the mail bot
The following pairs of cells are connected
c11 c12
c11 c21
c12 c12
c21 c11
c12 c13

```
c12 c22
c13 c12
c22 c12
c13 c23
c13 c14
c23 c13
c14 c13
c14 c15
c24 c14
c15 c14
c14 c24
c15 c25
c25 c15
c25 c24
c25 c35
c24 c25
c35 c25
c24 c34
c23 c24
c34 c24
c24 c23
c23 c33
c23 c22
c33 c23
c22 c23
c22 c32
c21 c22
c32 c22
c22 c21
c21 c31
c31 c21
c31 c32
c32 c33
c32 c31
c33 c32
c33 c43
c33 c34
c43 c33
c34 c33
c34 c35
c34 c44
c35 c34
c44 c34
c35 c45
```

c45 c35
c45 c55
c45 c44
c55 c45
c44 c45
c44 c54
c44 c43
c54 c44
c43 c44
c43 c53
c43 c42
c53 c43
c42 c43
c42 c52
c42 c41
c42 c42
c41 c42
c41 c51
c51 c41
c51 c52
c52 c51
c53 c54
c54 c53
c54 c55
c55 c54
END OF CONNECTED CELLS

inital state

(On p001 c11) and (On p010 c13) and (On p011 c15) and (On bot c33) and (On scanner c51) and (On switch c35) and (On bot c33) and (beltCell c44) and (beltCell c45)

Predicates
(Connected cellx , celly) true iff two cells connected
(BeltOn x) true iff delivery belt is on
(Scanned x) true iff package x is scanned
(HoldingObj bot ,y) true iff bit is holding object y
(On x , celly) true iff object x on celly
(Scanner x) true iff x is a scanner
(Package x) true iff x is a package
(Delivered x) true iff x is delivered
(beltCell x) true iff x is a belt cell
(Holding bot) true iff the bot is holding something

Task 1.2: Actions (10 marks)

```
action Move:
    parameters (mailBot x  y )
    precondition ( (On mailBot x) and (Connected x y) and (not (beltCell y)))
    effect ((On mailBot y) and (not (On mailBot x)))

action PickupPackage:
  parameters (mailBot  x y)
  precondition ( (On mailBot y) and (On x y) and (not( Holding mailBot)))
  effect ((HoldingObj mailBot x) and (not (On x y)) and (Holding mailBot))

  action PickupScanner
    parameters (mailBot x y )
    precondition ((On mailBot y) and (On x y) and (not (Holding mailBot)))
    effect ((HoldingObj mailBot x) and (not (On x y)) and (Holding mailBot)


action Scan
parameters (bot object cell scanner )
precondition (On bot cell) and (On object cell) and (HoldingObj bot scanner)
effect (Scanned object)

action turnOn
parameters (bot cell switch belt )
precondition ((On bot cell) and  (On switch cell)
effect (BeltOn belt))

action turnOff
parameters (bot cell switch belt )
precondition ( (On bot cell) and (On switch cell))
effect ((not (BeltOn belt)))



action PutDownPackage
parameters (mailBot x y)
precondition ((On mailBot y) and (HoldingObj mailbot x) and )
effect ((not(HoldingObj mailBot x)) and (On x y) and (not (Holding mailBot)))
```

action PutDownPackageBelt
parameters (mailBot x y z)
precondition ((On mailBot y) and (HoldingObj mailbot x) and (beltCell z) and (Connected y z)
effect ((not(HoldingObj mailBot x)) and (On x z) and (not (Holding mailBot)))

action PutDownScanner
parameters (mailBot x y)
precondition ((On mailBot y) and (HoldingObj mailbot x))
effect ((not(HoldingObj mailBot x)) and (On x y) and (not (Holding mailBot)))

action Deliver
parameters(package cell belt)
precondition)(BeltOn belt) and (On package cell) and (beltCell cell) and (scanned package))
effect ((Delivered package))

Task 1.3: Backwards state-space search (10 marks)

Ex backwards
▸ g1 = On(A, B) ∧ On(B, C)
▸ Available Actions: Move(A, x, B), Move(B, x, C)
▸ Choose : Move(A, x, B) ▸ g2 = On(A, x) ∧ Clear(A) ∧ Clear(B) ∧ Block(A) ∧ Block(B) ∧ A ⊭ x ∧ A⊭ B ∧ x⊭ B ∧ On(B, C)
▸ Available actions : Move(B, x ′ , C), Move(x ′ , B, y), Move(x ′ , A, y), Move(A, x ′ , x)
▸ Choose : Move(B, x ′ , C)
▸ g3 = On(A, x) ∧ Clear(A) ∧ Clear(B) ∧ Block(A) ∧ Block(B) ∧ A⊭ x ∧ A⊭ B ∧ x⊭ B ∧ On(B, x ′) ∧ Clear(B) ∧ Clear(C) ∧ Block(C) ∧ B⊭ x ′ ∧ B⊭ C ∧ x ′⊭ C
▸ g3 satisfies initial state by substituting {x = Table, x ′ = Table}
▸ Done! Final Plan: Move(B,Table, C), Move(A,Table, B)

Backwards Goal :

- g1 = Delivered (p011)
- Available actions : Deliver (P011 cella BELT)
- g2 = Choose Deliver (P011 cella BELT)
- Available actions : putDownPackageBelt (MAILBOT P011 cellb cella) , turnOn(MAILBOT SWITCH BELT)
- g3= choose   putDownPackageBelt (MAILBOT P011 cellb cella)
- Available actions : pickuppackage(MAILBOT P011 cellb)
- g4 = choose pickuppackage(MAILBOT P011 cellb)
- AvailableActions : Move(MAILBOT cellc cellb)

- g5 = choose Move(MAILBOT cellc cellb)
- Available actions : Move(MAILBOT celld cellc)
- g6=choose Move(celld cellc)
- Available actions : Move (celle celld)
- g7 = choose Move (celle celld)
- Plan is satisfied by substituting {cella= c44. Cellb = c54, cellc = c53, celld = c43,celle = c33}
- Final Plan : Move(BOT, C33, C43) Move(BOT, C43, C53), Move(BOT, C53, C54), Pickuppackage(BOT ,P011, C54), Putdownpackagebelt(BOT, P011, C54, C44) ,Deliver (P011, C44, BELT)

Task 2: Implementation (10 marks)

2.1 Task 2.1: Test Problem #1 (5 marks)

```
0: MOVE MAILBOT C33 C34
    1: MOVE MAILBOT C34 C35
    2: TURNON MAILBOT C35 SWITCH BELT
    3: MOVE MAILBOT C35 C34
    4: MOVE MAILBOT C34 C33
    5: MOVE MAILBOT C33 C43
    6: MOVE MAILBOT C43 C42
    7: MOVE MAILBOT C42 C41
    8: MOVE MAILBOT C41 C51
    9: PICKUPSCANNER MAILBOT SCANNER C51
   10: MOVE MAILBOT C51 C41
   11: MOVE MAILBOT C41 C42
   12: MOVE MAILBOT C42 C43
   13: MOVE MAILBOT C43 C33
   14: MOVE MAILBOT C33 C23
   15: MOVE MAILBOT C23 C22
   16: MOVE MAILBOT C22 C21
   17: MOVE MAILBOT C21 C11
   18: SCAN MAILBOT P001 C11 SCANNER
   19: PUTDOWNSCANNER MAILBOT SCANNER C11
   20: PICKUPPACKAGE MAILBOT P001 C11
```

```
      21: MOVE MAILBOT C11 C12
      22: MOVE MAILBOT C12 C13
      23: MOVE MAILBOT C13 C14
      24: MOVE MAILBOT C14 C24
      25: MOVE MAILBOT C24 C34
      26: PUTDOWNPACKAGEBELT MAILBOT P001 C34 C44
      27: DELIVER P001 C44 BELT
```

Task 2.2: Test Problem #2 (5 marks)

```
 0: MOVE MAILBOT C33 C34
       1: MOVE MAILBOT C34 C35
       2: TURNON MAILBOT C35 SWITCH BELT
       3: MOVE MAILBOT C35 C34
       4: MOVE MAILBOT C34 C33
       5: MOVE MAILBOT C33 C43
       6: MOVE MAILBOT C43 C42
       7: MOVE MAILBOT C42 C41
       8: MOVE MAILBOT C41 C51
       9: PICKUPSCANNER MAILBOT SCANNER C51
      10: MOVE MAILBOT C51 C41
      11: MOVE MAILBOT C41 C42
      12: MOVE MAILBOT C42 C43
      13: MOVE MAILBOT C43 C33
      14: MOVE MAILBOT C33 C23
      15: MOVE MAILBOT C23 C22
      16: MOVE MAILBOT C22 C21
      17: MOVE MAILBOT C21 C11
      18: SCAN MAILBOT P001 C11 SCANNER
      19: PUTDOWNSCANNER MAILBOT SCANNER C11
      20: PICKUPPACKAGE MAILBOT P001 C11
      21: MOVE MAILBOT C11 C12
      22: MOVE MAILBOT C12 C13
      23: MOVE MAILBOT C13 C14
      24: MOVE MAILBOT C14 C24
      25: MOVE MAILBOT C24 C34
      26: PUTDOWNPACKAGEBELT MAILBOT P001 C34 C44
      27: DELIVER P001 C44 BELT
```

Task 3: Experiment (15 marks)

Task 3.1: Design (5 marks)

In order to make the problem harder I had a total of 5 packages around the warehouse all requiring be delivered, i made the belt only one cell. I disconnected 2 cells to create a wall near the belt , hence more branching, and I required the bot and scanner to end up where they started.

Task 3.2: Evaluation (10 marks)

Given the problem-3.pddl, design an experiment and evaluate the effect of different values of w and g

1.
G = 1 h = 5

```
    4.81 seconds searching, evaluating 48198 states, to a max depth of 0
        4.81 seconds total tim
```

2.
G = 5, h = 1
Did not terminate in a reasonable time frame on dice machine (> 15min)
3. G = 3 , h = 3
Did not terminate in a reasonable time frame on dice machine (> 15min)

4. G = 10 . h =100

```
       0.14 seconds searching, evaluating 4662 states, to a max depth of 0
        0.14 seconds total time
```

5. G = 100 h = 10
Did not terminate in a reasonable time frame on dice machine (> 15min)

6. G = 1 h = 10

```
s, 0 relevant fluents
        0.00 seconds computing LNF
        0.00 seconds building connectivity graph
        0.15 seconds searching, evaluating 4662 states, to a max depth of 0
        0.15 seconds total
```

7. G =2 h =4
Did not terminate in a reasonable time frame on dice machine (> 15min)

8. 1*g(s) + 100000*h(s)
0.15 seconds total
4668

10*g(s) + 50*h(s)

8.76 seconds searching, evaluating 48198 states, to a max depth of 0

After running these experiments I have found that we need h to wait higher we require h to be a factor larger than g. All experiments where the value of h was at least a factor of 10 greater than g performed well i.e 1*g(s) + 100000*h(s) , 0.15 seconds total , 4668
, G = 1 h = 10 , 0.15 seconds searching, evaluating 4662 states, G = 10 . h =100, 0.14 seconds searching, evaluating 4662 states.
We see here that g = 1 h = 10 and g = 10, h = 100 are almost identical which is what we would expect given that the scaling factor g to h is identical, the discrepancy in time is likely situational to the current state of the machine at run time. I experimented to see if a larger scaling factor from g to h consistently meant improved results but running with g = 1 , h = 100000, gave 0.15 seconds and 6 more states so a slight decrease. Further halving the factor from 10 to 5 and running g = 10 h = 50 and g = 1, h =5 , gave significantly worse results with times 8.76 seconds, 48198 states, 4.81 seconds and 48198 states respectively. G =2 h =4 did not run in a reasonable time frame even when left to run on a dice machine for over 15 minutes so here we see that only values with h is at least 5 x bigger work well. # Could try a few others and try converge on an optimal value.

Task 4.1: Energy Station (10 marks)

```
 0: MOVE MAILBOT C33 C32
 1: MOVE MAILBOT C32 C31
 2: RECHARGE MAILBOT C31
 3: MOVE MAILBOT C31 C32
 4: MOVE MAILBOT C32 C33
 5: MOVE MAILBOT C33 C43
 6: MOVE MAILBOT C43 C42
 7: MOVE MAILBOT C42 C41
 8: MOVE MAILBOT C41 C51
 9: PICKUPSCANNER MAILBOT SCANNER C51
10: MOVEHOLDING MAILBOT C51 C41
11: MOVEHOLDING MAILBOT C41 C42
12: MOVEHOLDING MAILBOT C42 C43
13: PUTDOWNSCANNER MAILBOT SCANNER C43
```

```
14: MOVE MAILBOT C43 C33
15: MOVE MAILBOT C33 C32
16: MOVE MAILBOT C32 C31
17: RECHARGE MAILBOT C31
18: MOVE MAILBOT C31 C32
19: MOVE MAILBOT C32 C33
20: MOVE MAILBOT C33 C43
21: PICKUPSCANNER MAILBOT SCANNER C43
22: MOVEHOLDING MAILBOT C43 C33
23: MOVEHOLDING MAILBOT C33 C34
24: MOVEHOLDING MAILBOT C34 C35
25: TURNON MAILBOT C35 SWITCH BELT
26: MOVEHOLDING MAILBOT C35 C34
27: PUTDOWNSCANNER MAILBOT SCANNER C34
28: MOVE MAILBOT C34 C33
29: MOVE MAILBOT C33 C32
30: MOVE MAILBOT C32 C31
31: RECHARGE MAILBOT C31
32: MOVE MAILBOT C31 C21
33: MOVE MAILBOT C21 C11
34: PICKUPPACKAGE MAILBOT P001 C11
35: MOVEHOLDING MAILBOT C11 C12
36: PUTDOWNPACKAGE MAILBOT P001 C12
37: MOVE MAILBOT C12 C13
38: PICKUPPACKAGE MAILBOT P010 C13
39: MOVEHOLDING MAILBOT C13 C23
40: MOVEHOLDING MAILBOT C23 C33
41: MOVEHOLDING MAILBOT C33 C34
42: PUTDOWNPACKAGE MAILBOT P010 C34
43: PICKUPSCANNER MAILBOT SCANNER C34
44: SCAN MAILBOT P010 C34 SCANNER
45: PUTDOWNSCANNER MAILBOT SCANNER C34
46: PICKUPPACKAGE MAILBOT P010 C34
47: PUTDOWNPACKAGEBELT MAILBOT P010 C34 C44
48: DELIVER P010 C44 BELT MAILBOT
49: MOVE MAILBOT C34 C33
50: MOVE MAILBOT C33 C32
51: MOVE MAILBOT C32 C31
52: RECHARGE MAILBOT C31
53: MOVE MAILBOT C31 C32
54: MOVE MAILBOT C32 C22
55: MOVE MAILBOT C22 C12
56: PICKUPPACKAGE MAILBOT P001 C12
57: MOVEHOLDING MAILBOT C12 C22
```

```
      58: MOVEHOLDING MAILBOT C22 C32
      59: MOVEHOLDING MAILBOT C32 C33
      60: MOVEHOLDING MAILBOT C33 C34
      61: PUTDOWNPACKAGE MAILBOT P001 C34
      62: PICKUPSCANNER MAILBOT SCANNER C34
      63: SCAN MAILBOT P001 C34 SCANNER
      64: PUTDOWNSCANNER MAILBOT SCANNER C34
      65: PICKUPPACKAGE MAILBOT P001 C34
      66: PUTDOWNPACKAGEBELT MAILBOT P001 C34 C44
      67: DELIVER P001 C44 BELT MAILBOT
```

Task 4.2: Limited Power (15 marks)

```
step    0: MOVE BOT C33 C34
        1: MOVE BOT C34 C35
        2: TURNON BOT C35 SWITCH BELT
        3: MOVE DELIVERYBOT C55 C54
        4: MOVE BOT C35 C25
        5: MOVE BOT C25 C15
        6: MOVE DELIVERYBOT C54 C53
        7: MOVE DELIVERYBOT C53 C43
        8: MOVE DELIVERYBOT C43 C33
        9: MOVE DELIVERYBOT C33 C32
       10: MOVE DELIVERYBOT C32 C31
       11: RECHARGE DELIVERYBOT C31
       12: MOVE DELIVERYBOT C31 C32
       13: MOVE DELIVERYBOT C32 C33
       14: MOVE DELIVERYBOT C33 C34
       15: MOVE DELIVERYBOT C34 C35
       16: MOVE DELIVERYBOT C35 C25
       17: MOVE DELIVERYBOT C25 C15
       18: PICKUPHEAVYPACKAGE BOT DELIVERYBOT P011 C15
       19: MOVEHOLDINGHEAVY BOT C15 C25 DELIVERYBOT
       20: MOVEHOLDINGHEAVY BOT C25 C35 DELIVERYBOT
       21: PUTDOWNPACKAGEHEAVY BOT DELIVERYBOT P011 C35
       22: MOVE DELIVERYBOT C35 C34
       23: MOVE DELIVERYBOT C34 C33
       24: MOVE DELIVERYBOT C33 C32
       25: MOVE DELIVERYBOT C32 C31
       26: RECHARGE DELIVERYBOT C31
       27: MOVE DELIVERYBOT C31 C32
       28: MOVE DELIVERYBOT C32 C33
```

```
        29: MOVE DELIVERYBOT C33 C43
        30: MOVE DELIVERYBOT C43 C42
        31: MOVE DELIVERYBOT C42 C41
        32: MOVE DELIVERYBOT C41 C51
        33: PICKUPSCANNER DELIVERYBOT SCANNER C51
        34: MOVEHOLDING DELIVERYBOT C51 C41
        35: MOVEHOLDING DELIVERYBOT C41 C42
        36: MOVEHOLDING DELIVERYBOT C42 C43
        37: PUTDOWNSCANNER DELIVERYBOT SCANNER C43
        38: MOVE DELIVERYBOT C43 C33
        39: MOVE DELIVERYBOT C33 C32
        40: MOVE DELIVERYBOT C32 C31
        41: RECHARGE DELIVERYBOT C31
        42: MOVE DELIVERYBOT C31 C32
        43: MOVE DELIVERYBOT C32 C33
        44: MOVE DELIVERYBOT C33 C43
        45: PICKUPSCANNER DELIVERYBOT SCANNER C43
        46: MOVEHOLDING DELIVERYBOT C43 C33
        47: MOVEHOLDING DELIVERYBOT C33 C34
        48: MOVEHOLDING DELIVERYBOT C34 C35
        49: SCAN DELIVERYBOT P011 C35 SCANNER
        50: PUTDOWNSCANNER DELIVERYBOT SCANNER C35
        51: PICKUPHEAVYPACKAGE BOT DELIVERYBOT P011 C35
        52: PUTDOWNPACKAGEHEAVYBELT BOT DELIVERYBOT P011 C35 C45
        53: DELIVER P011 C45 BELT DELIVERYBOT


time spent:    0.00 seconds instantiating 1350 easy, 0 hard action templates
               0.00 seconds reachability analysis, yielding 110 facts and 632 actions
               0.00 seconds creating final representation with 109 relevant facts, 2 relevant fluents
               0.00 seconds computing LNF
               0.00 seconds building connectivity graph
               2.90 seconds searching, evaluating 49748 states, to a max depth of 0
               2.90 seconds total time
```

Task 4.3: Your Extension (25 marks)

I will allow 3 classes of packages

Royal mail special delivery
Royal mail 1st class
Standard package.

Special delivery packages must be delivered before 1st class, 1st class before the second class.

Extending domain :

Types :
Priority # This allows us to have a numerical fluent value for the number of packages left

New predicates :
(firstClass ?x - package) #ture iff a package is first class
(special ?x - package) #true iff a package is special class

Functions :
(firstClassRemaining ?first - priority) #stores as an int the number of remaining first-class packages
(specialRemaining ?special - priority) #stores as an int the number of remaining special class packages

Actions:

```
(:action Deliver
        :parameters(?package - package ?cell - cell ?belt - belt ?mailBot - bot  ?first - priority
?special - priority)
        :precondition(and
          (not (firstClass ?package))
          (< (firstClassRemaining ?first) 1)
          (< (specialRemaining ?special) 1)
          (BeltOn ?belt)
          (On ?package ?cell)
          (beltCell ?cell)
          (scanned ?package)
            (> (battery-amount ?mailBot) 0) )
       :effect (and
            (Delivered ?package)
            ))
```

This action was changed to make sure that there were no remaining special or first class delivery packages remaining, and the package trying to be delivered is not special or first class.

```
            (:action DeliverFirst
              :parameters(?package - package ?cell - cell ?belt - belt ?mailBot - bot  ?first - priority
?special - priority)
:precondition(and
  (< (specialRemaining ?special) 1)
  (BeltOn ?belt)
  (On ?package ?cell)
  (beltCell ?cell)
  (scanned ?package)
  (firstClass ?package)
    (> (battery-amount ?mailBot) 0) )
:effect (and
    (Delivered ?package)
    (decrease (firstClassRemaining ?first) 1)
    ))
```

This action was introduced to allow first class packages to be delivered, it only has the preconditions that there must be no special packages remaining and the package is first class/ It also has an additional effect that the amount of first class remaining packages is reduced by 1 since 1 has been delivered.

```
(:action Deliverspecial
  :parameters(?package - package ?cell - cell ?belt - belt ?mailBot - bot    ?special - priority)
  :precondition(and

    (BeltOn ?belt)
    (On ?package ?cell)
    (beltCell ?cell)
    (scanned ?package)
    (special ?package)
     (> (battery-amount ?mailBot) 0) )
  :effect (and
      (Delivered ?package)
      (decrease (specialRemaining ?special) 1)
      ))
```

This action was introduced to allow special class packages to be delivered, it only has the preconditions that delivery before my own extension had plus checking the package is special with the predicate special. It has the effect of delivering the package but also decreasing the number of remaining special packages.

PLanning Problem for my extension I had 4 packages, i specified locations and delivered in goal state as with previous plans. I stated their class with
(firstClass f001)
(special s010)
(special s000)
The 4th package p011 is a standard package for which their is no predicate.

I then told the planner the numbers of these packages for the numerical fluents

```
  (= (battery-amount mailBot) 10)
   (= (firstClassRemaining firstClassAm) 1)
    (= (specialRemaining specialClassAm) 2)
```

As well as adding to the objects :

```
firstClassAm - priority
specialAm - priority
```

```
0: MOVE MAILBOT C33 C34
       1: MOVE MAILBOT C34 C35
       2: TURNON MAILBOT C35 SWITCH BELT
       3: MOVE MAILBOT C35 C34
       4: MOVE MAILBOT C34 C33
       5: MOVE MAILBOT C33 C32
       6: MOVE MAILBOT C32 C31
       7: RECHARGE MAILBOT C31
       8: MOVE MAILBOT C31 C32
       9: MOVE MAILBOT C32 C33
      10: MOVE MAILBOT C33 C43
      11: MOVE MAILBOT C43 C42
      12: MOVE MAILBOT C42 C41
      13: MOVE MAILBOT C41 C51
      14: PICKUPSCANNER MAILBOT SCANNER C51
      15: MOVEHOLDING MAILBOT C51 C41
      16: MOVEHOLDING MAILBOT C41 C42
      17: MOVEHOLDING MAILBOT C42 C43
      18: PUTDOWNSCANNER MAILBOT SCANNER C43
      19: MOVE MAILBOT C43 C33
      20: MOVE MAILBOT C33 C32
      21: MOVE MAILBOT C32 C31
      22: RECHARGE MAILBOT C31
      23: MOVE MAILBOT C31 C32
      24: MOVE MAILBOT C32 C33
      25: MOVE MAILBOT C33 C43
      26: PICKUPSCANNER MAILBOT SCANNER C43
      27: MOVEHOLDING MAILBOT C43 C33
      28: MOVEHOLDING MAILBOT C33 C23
      29: MOVEHOLDING MAILBOT C23 C22
      30: MOVEHOLDING MAILBOT C22 C21
      31: MOVEHOLDING MAILBOT C21 C11
      32: SCAN MAILBOT F001 C11 SCANNER
      33: SCAN MAILBOT S000 C11 SCANNER
      34: PUTDOWNSCANNER MAILBOT SCANNER C11
```

```
35: MOVE MAILBOT C11 C21
36: MOVE MAILBOT C21 C31
37: RECHARGE MAILBOT C31
38: MOVE MAILBOT C31 C21
39: MOVE MAILBOT C21 C11
40: PICKUPSCANNER MAILBOT SCANNER C11
41: MOVEHOLDING MAILBOT C11 C12
42: MOVEHOLDING MAILBOT C12 C13
43: SCAN MAILBOT S010 C13 SCANNER
44: PUTDOWNSCANNER MAILBOT SCANNER C13
45: PICKUPPACKAGE MAILBOT S010 C13
46: MOVEHOLDING MAILBOT C13 C23
47: MOVEHOLDING MAILBOT C23 C33
48: MOVEHOLDING MAILBOT C33 C43
49: PUTDOWNPACKAGEBELT MAILBOT S010 C43 C44
50: DELIVERSPECIAL S010 C44 BELT MAILBOT SPECIALAM
51: MOVE MAILBOT C43 C33
52: MOVE MAILBOT C33 C32
53: MOVE MAILBOT C32 C31
54: RECHARGE MAILBOT C31
55: MOVE MAILBOT C31 C21
56: MOVE MAILBOT C21 C11
57: PICKUPPACKAGE MAILBOT S000 C11
58: MOVEHOLDING MAILBOT C11 C12
59: MOVEHOLDING MAILBOT C12 C13
60: MOVEHOLDING MAILBOT C13 C23
61: MOVEHOLDING MAILBOT C23 C33
62: MOVEHOLDING MAILBOT C33 C43
63: PUTDOWNPACKAGEBELT MAILBOT S000 C43 C44
64: DELIVERSPECIAL S000 C44 BELT MAILBOT SPECIALAM
65: MOVE MAILBOT C43 C33
66: MOVE MAILBOT C33 C32
67: MOVE MAILBOT C32 C31
68: RECHARGE MAILBOT C31
69: MOVE MAILBOT C31 C21
70: MOVE MAILBOT C21 C11
71: PICKUPPACKAGE MAILBOT F001 C11
72: MOVEHOLDING MAILBOT C11 C12
73: PUTDOWNPACKAGE MAILBOT F001 C12
74: MOVE MAILBOT C12 C13
75: MOVE MAILBOT C13 C14
76: MOVE MAILBOT C14 C15
77: PICKUPPACKAGE MAILBOT P011 C15
78: MOVEHOLDING MAILBOT C15 C14
```

```
 79: MOVEHOLDING MAILBOT C14 C13
 80: PUTDOWNPACKAGE MAILBOT P011 C13
 81: PICKUPSCANNER MAILBOT SCANNER C13
 82: SCAN MAILBOT P011 C13 SCANNER
 83: PUTDOWNSCANNER MAILBOT SCANNER C13
 84: MOVE MAILBOT C13 C23
 85: MOVE MAILBOT C23 C33
 86: MOVE MAILBOT C33 C32
 87: MOVE MAILBOT C32 C31
 88: RECHARGE MAILBOT C31
 89: MOVE MAILBOT C31 C32
 90: MOVE MAILBOT C32 C33
 91: MOVE MAILBOT C33 C23
 92: MOVE MAILBOT C23 C13
 93: PICKUPPACKAGE MAILBOT P011 C13
 94: MOVEHOLDING MAILBOT C13 C23
 95: MOVEHOLDING MAILBOT C23 C33
 96: MOVEHOLDING MAILBOT C33 C43
 97: PUTDOWNPACKAGEBELT MAILBOT P011 C43 C44
 98: MOVE MAILBOT C43 C33
 99: MOVE MAILBOT C33 C32
100: MOVE MAILBOT C32 C31
101: RECHARGE MAILBOT C31
102: MOVE MAILBOT C31 C32
103: MOVE MAILBOT C32 C22
104: MOVE MAILBOT C22 C12
105: PICKUPPACKAGE MAILBOT F001 C12
106: MOVEHOLDING MAILBOT C12 C22
107: MOVEHOLDING MAILBOT C22 C32
108: MOVEHOLDING MAILBOT C32 C33
109: MOVEHOLDING MAILBOT C33 C43
110: PUTDOWNPACKAGEBELT MAILBOT F001 C43 C44
111: DELIVERFIRST F001 C44 BELT MAILBOT FIRSTCLASSAM SPECIALAM
112: DELIVER P011 C44 BELT MAILBOT FIRSTCLASSAM SPECIALAM
```

As can be seen the bot successfully delivers packages in priority order however it does try to deal with packages that are not priority such as moving them onto or closer to the belt to fix this i could prevent it from picking up non priority packages until all priority packages had been delivered.