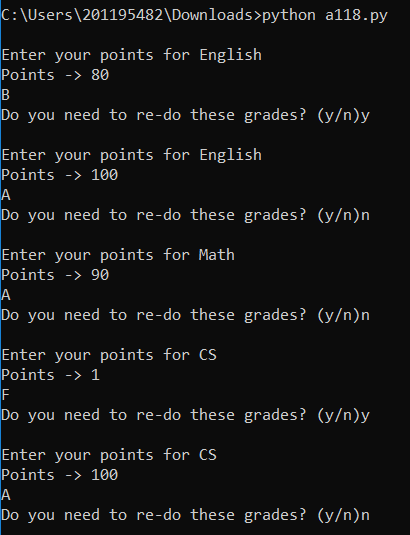
CSP 1.1.8 - Turtles in Traffic

Kyle Huang - CSP Period 5

[Link to Google Doc (If reading from PDF)](https://docs.google.com/document/d/1Vtxme5iK5PFIKycTv10-S7ltZx3qaV_FrVK1hv9QJn8/edit?usp=sharing)

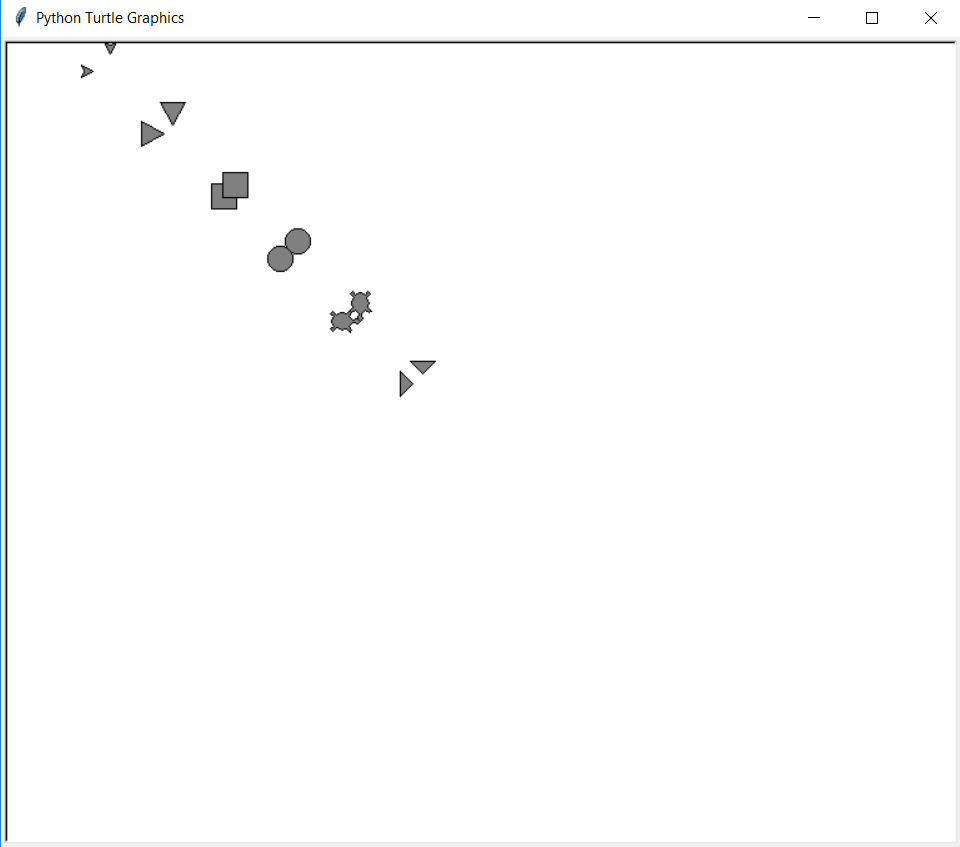
17. (17 on PLTW, but not on canvas assignment)

| # a118\_grades.py  my\_courses = ["English", "Math", "CS"]  for course in my\_courses:  redo = "y"  while (redo == "y"):  print() # blank line  print("Enter your points for " + course)  points = int(input("Points -> "))    if (points >= 90):  print("A")  elif (points >= 80):  print("B")  elif (points >= 70):  print("C")  elif (points >= 60):  print("D")  else:  print("F")    redo = input("Do you need to re-do these grades? (y/n)") |
| --- |



20. (17 on canvas, but 20 on PLTW)

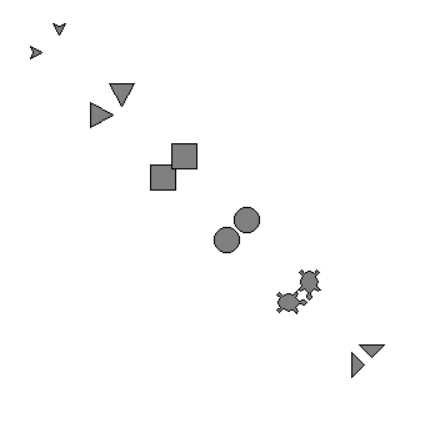
| # a118\_turtles\_in\_traffic.py  # Move turtles horizontally and vertically across screen.  # Stopping turtles when they collide.  import turtle as trtl  # create two empty lists of turtles, adding to them later  horiz\_turtles = []  vert\_turtles = []  # use interesting shapes and colors  turtle\_shapes = ["arrow", "turtle", "circle", "square", "triangle", "classic"]  horiz\_colors = ["red", "blue", "green", "orange", "purple", "gold"]  vert\_colors = ["darkred", "darkblue", "lime", "salmon", "indigo", "brown"]  tloc = 50  for s in turtle\_shapes:  ht = trtl.Turtle(shape=s)  horiz\_turtles.append(ht)  ht.penup()  new\_color = horiz\_colors.pop()  ht.fillcolor(new\_color)  ht.goto(-350, tloc)  ht.setheading(0)  vt = trtl.Turtle(shape=s)  vert\_turtles.append(vt)  vt.penup()  new\_color = vert\_colors.pop()  vt.fillcolor(new\_color)  vt.goto( -tloc, 350)  vt.setheading(270)  tloc += 50  # TODO: move turtles across and down screen, stopping for collisions  steps = 0  while steps < 50:  for ht in horiz\_turtles:  collision = False  for vt in vert\_turtles:  ht.forward(3)  vt.forward(3)  if (abs(ht.xcor()-vt.xcor()) <= 20 and abs(ht.ycor()-vt.ycor()) <= 20):  if (not collision):  horiz\_turtles.remove(ht)  ht.fillcolor("gray")  vert\_turtles.remove(vt)  vt.fillcolor("gray")  collision = True  steps = steps + 1  wn = trtl.Screen()  wn.mainloop() |
| --- |



21. (18 on canvas, but 21 on PLTW)

| # a118\_turtles\_in\_traffic.py  # Move turtles horizontally and vertically across screen.  # Stopping turtles when they collide.  import turtle as trtl  # create two empty lists of turtles, adding to them later  horiz\_turtles = []  vert\_turtles = []  # use interesting shapes and colors  turtle\_shapes = ["arrow", "turtle", "circle", "square", "triangle", "classic"]  horiz\_colors = ["red", "blue", "green", "orange", "purple", "gold"]  vert\_colors = ["darkred", "darkblue", "lime", "salmon", "indigo", "brown"]  tloc = 50  for s in turtle\_shapes:  ht = trtl.Turtle(shape=s)  horiz\_turtles.append(ht)  ht.penup()  new\_color = horiz\_colors.pop()  ht.fillcolor(new\_color)  ht.goto(-350, tloc)  ht.setheading(0)  vt = trtl.Turtle(shape=s)  vert\_turtles.append(vt)  vt.penup()  new\_color = vert\_colors.pop()  vt.fillcolor(new\_color)  vt.goto( -tloc, 350)  vt.setheading(270)  tloc += 50  # TODO: move turtles across and down screen, stopping for collisions  steps = 0  while steps < 50:  for ht in horiz\_turtles:  collision = False  for vt in vert\_turtles:  ht.forward(steps%12)  vt.forward(steps%12)  if (abs(ht.xcor()-vt.xcor()) <= 20 and abs(ht.ycor()-vt.ycor()) <= 20):  if (not collision):  horiz\_turtles.remove(ht)  ht.fillcolor("gray")  vert\_turtles.remove(vt)  vt.fillcolor("gray")  collision = True  steps = steps + 1  wn = trtl.Screen()  wn.mainloop() |
| --- |

Note: the turtles still crash at the same spots, but get there faster.

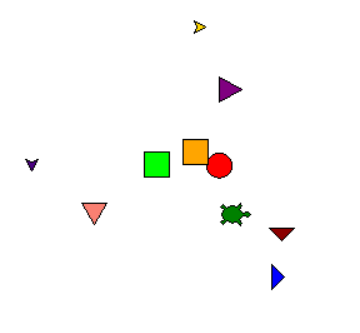


22. (19 on canvas, but 22 on PLTW)

[python file](https://drive.google.com/file/d/12twBogmOzZ1YkAxs4GUX8QPqXo-gJglu/view?usp=sharing)

| # a118\_turtles\_in\_traffic.py  # Move turtles horizontally and vertically across screen.  # Stopping turtles when they collide.  import turtle as trtl  # create two empty lists of turtles, adding to them later  horiz\_turtles = []  vert\_turtles = []  # use interesting shapes and colors  turtle\_shapes = ["arrow", "turtle", "square", "triangle", "classic"]  horiz\_colors = ["blue", "green", "orange", "purple", "gold"]  vert\_colors = ["darkred", "darkblue", "lime", "salmon", "indigo"]  crashed\_turtles = []  tloc = 50  for i in range(len(turtle\_shapes)):  s = turtle\_shapes[i]  ht = trtl.Turtle(shape=s)  horiz\_turtles.append(ht)  ht.penup()  new\_color = horiz\_colors[i]  ht.fillcolor(new\_color)  ht.goto(-350, tloc)  ht.setheading(0)  vt = trtl.Turtle(shape=s)  vert\_turtles.append(vt)  vt.penup()  new\_color = vert\_colors[i]  vt.fillcolor(new\_color)  vt.goto( -tloc, 350)  vt.setheading(270)  tloc += 50  # TODO: move turtles across and down screen, stopping for collisions  steps = 5  while steps < 55:  for ht in horiz\_turtles:  for vt in vert\_turtles:  speed = ((steps%10)-5)\*((steps%10)-5)/3  if (crashed\_turtles.\_\_contains\_\_(ht)):  ht.forward(max(-speed\*4,-30))  ht.shape(turtle\_shapes[horiz\_turtles.index(ht)])  ht.fillcolor(horiz\_colors[horiz\_turtles.index(ht)])  crashed\_turtles.remove(ht)  if (crashed\_turtles.\_\_contains\_\_(vt)):  vt.forward(max(-speed\*6,-45))  vt.shape(turtle\_shapes[vert\_turtles.index(vt)])  vt.fillcolor(vert\_colors[vert\_turtles.index(vt)])  crashed\_turtles.remove(vt)  if (ht.xcor() <= 200):  ht.forward(speed)  if (vt.ycor() >= -200):  vt.forward(speed)  if (abs(ht.xcor()-vt.xcor()) <= 20 and abs(ht.ycor()-vt.ycor()) <= 20):  ht.shape("circle")  ht.fillcolor("red")  vt.shape("circle")  vt.fillcolor("red")  crashed\_turtles.append(ht)  crashed\_turtles.append(vt)  steps = steps + 1  # End of program notifier  for i in range(len(horiz\_turtles)):  horiz\_turtles[i].color("gray")  vert\_turtles[i].color("gray")  wn = trtl.Screen()  wn.mainloop() |
| --- |

During a collision:



End of Program:

