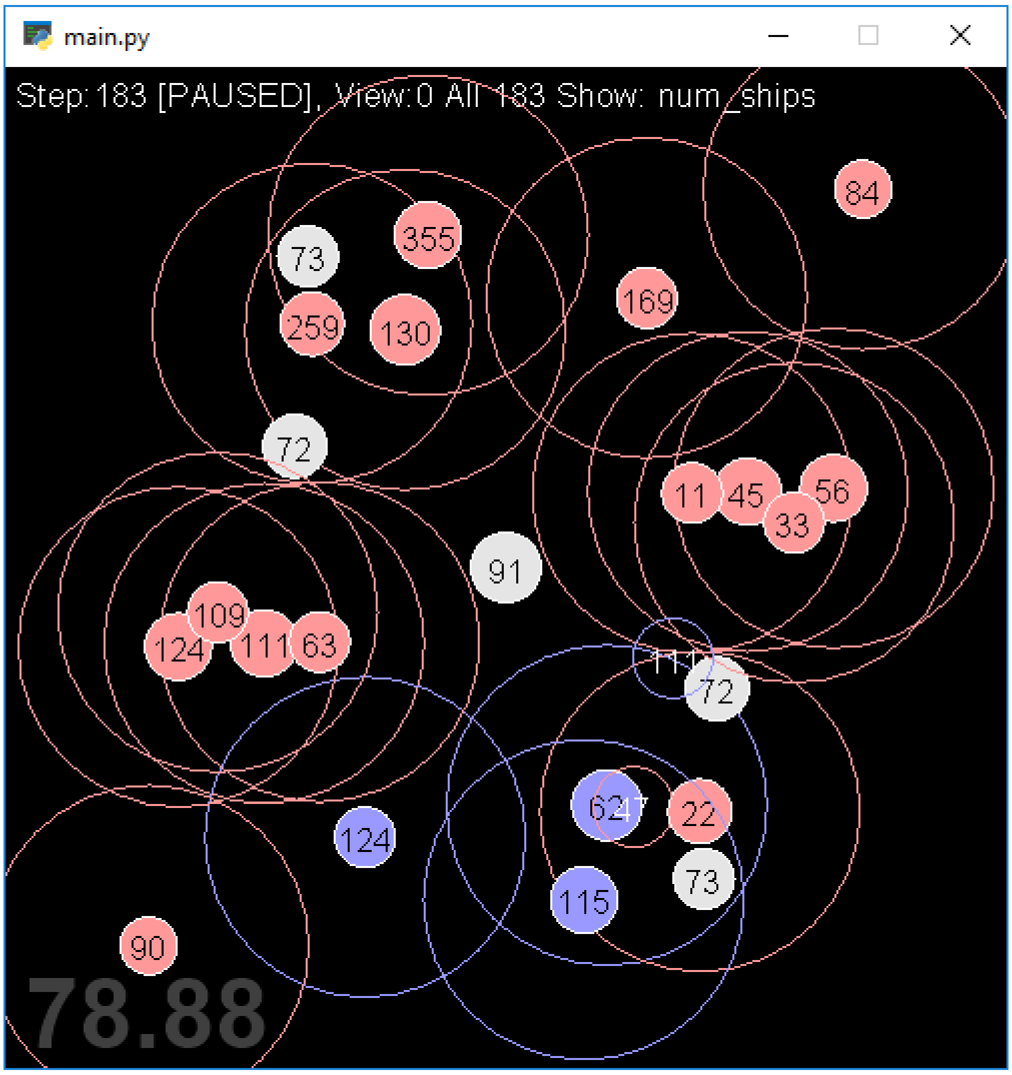
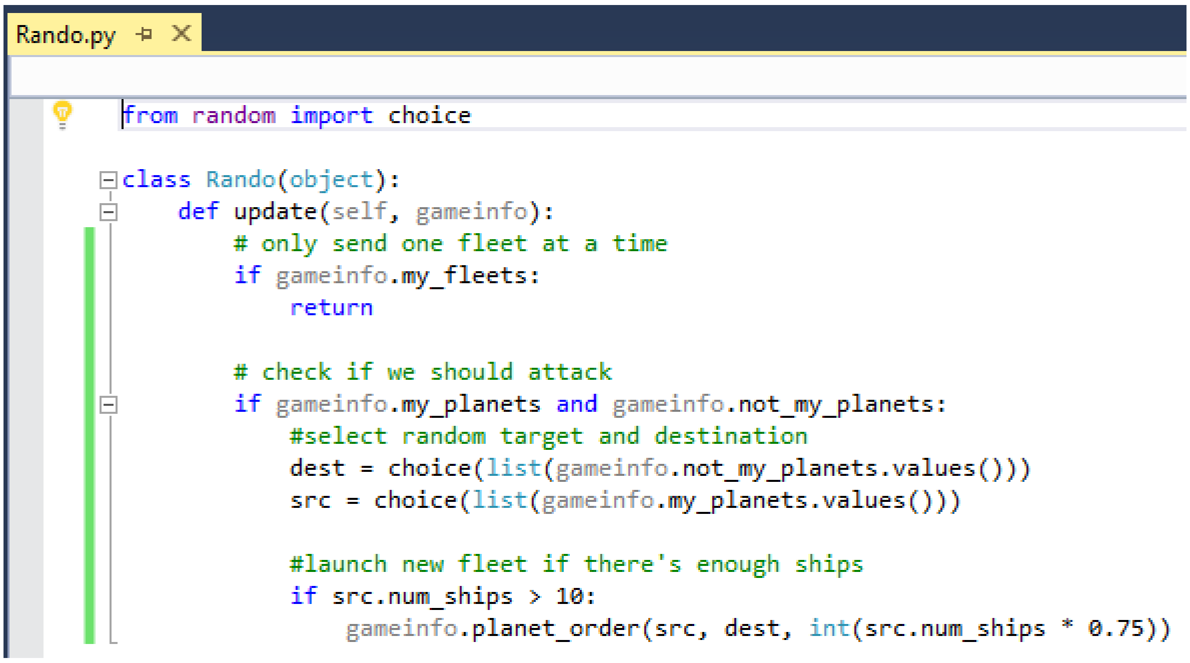
PlanetWars AI Pseudocode

Game Screenshot



Rando

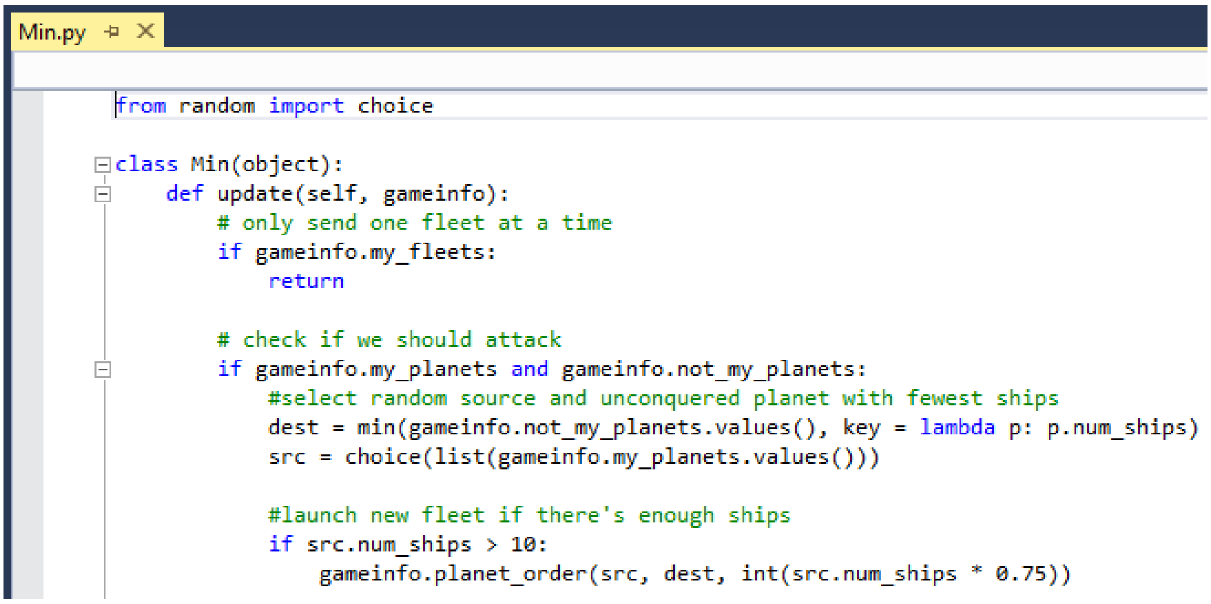
```
Update():
    If there's already a fleet out:
        Do nothing

    If I have planet(s) and there are planets I don't control:
        Dest = random planet I don't control
        Src = random planet I control

        If Src has more than 10 ships:
            Send 75% of ships from Src to Dest
```



```python
from random import choice

class Rando(object):
    def update(self, gameinfo):
        # only send one fleet at a time
        if gameinfo.my_fleets:
            return

        # check if we should attack
        if gameinfo.my_planets and gameinfo.not_my_planets:
            #select random target and destination
            dest = choice(list(gameinfo.not_my_planets.values()))
            src = choice(list(gameinfo.my_planets.values()))

            #launch new fleet if there's enough ships
            if src.num_ships > 10:
                gameinfo.planet_order(src, dest, int(src.num_ships * 0.75))
```

Min

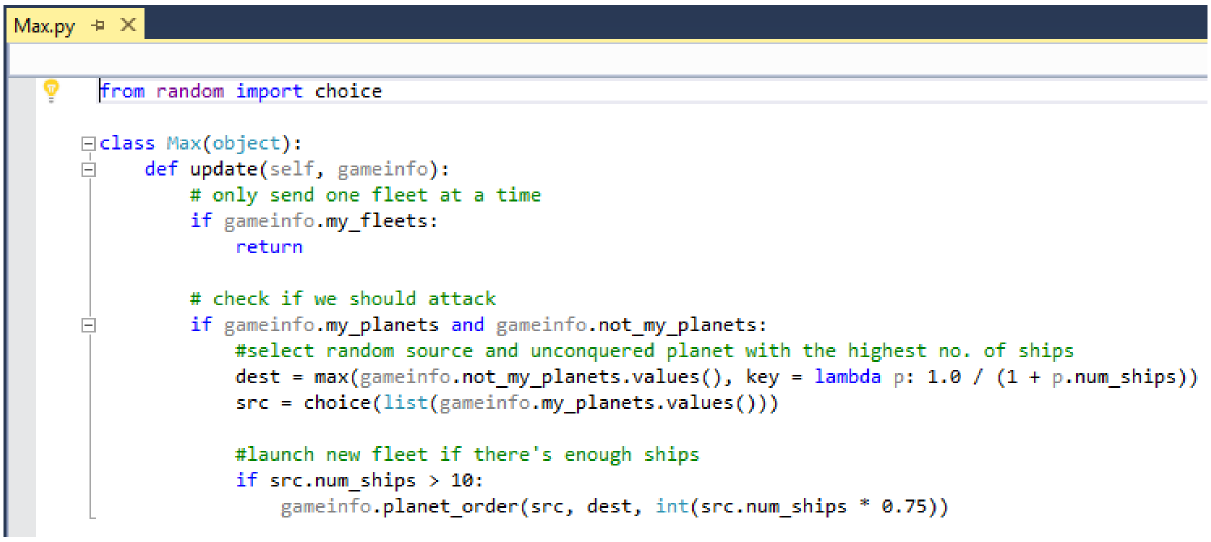
```
Update():
    If there's already a fleet out:
        Do nothing

    If I have planet(s) and there are planets I don't control:
        Dest = planet I don't control with the fewest ships
        Src = random planet I control

        If Src has more than 10 ships:
            Send 75% of ships from Src to Dest
```



```python
from random import choice

class Min(object):
    def update(self, gameinfo):
        # only send one fleet at a time
        if gameinfo.my_fleets:
            return

        # check if we should attack
        if gameinfo.my_planets and gameinfo.not_my_planets:
            #select random source and unconquered planet with fewest ships
            dest = min(gameinfo.not_my_planets.values(), key = lambda p: p.num_ships)
            src = choice(list(gameinfo.my_planets.values()))

            #launch new fleet if there's enough ships
            if src.num_ships > 10:
                gameinfo.planet_order(src, dest, int(src.num_ships * 0.75))
```

Max

```
Update():
        If there's already a fleet out:
                Do nothing

        If I have planet(s) and there are planets I don't control:
                Dest = planet I don't control with the most ships
                Src = random planet I control

                If Src has more than 10 ships:
                        Send 75% of ships from Src to Dest
```

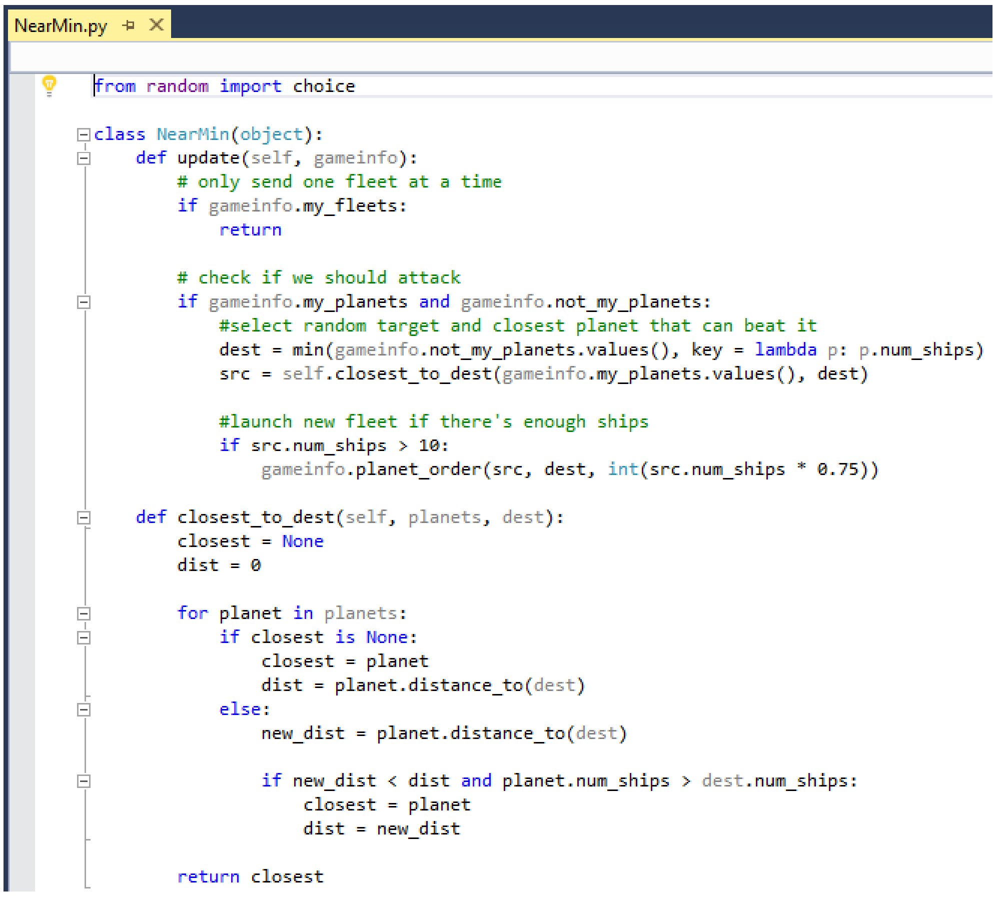


Max.py

```python
from random import choice

class Max(object):
    def update(self, gameinfo):
        # only send one fleet at a time
        if gameinfo.my_fleets:
            return

        # check if we should attack
        if gameinfo.my_planets and gameinfo.not_my_planets:
            #select random source and unconquered planet with the highest no. of ships
            dest = max(gameinfo.not_my_planets.values(), key = lambda p: 1.0 / (1 + p.num_ships))
            src = choice(list(gameinfo.my_planets.values()))

            #launch new fleet if there's enough ships
            if src.num_ships > 10:
                gameinfo.planet_order(src, dest, int(src.num_ships * 0.75))
```

NearMin

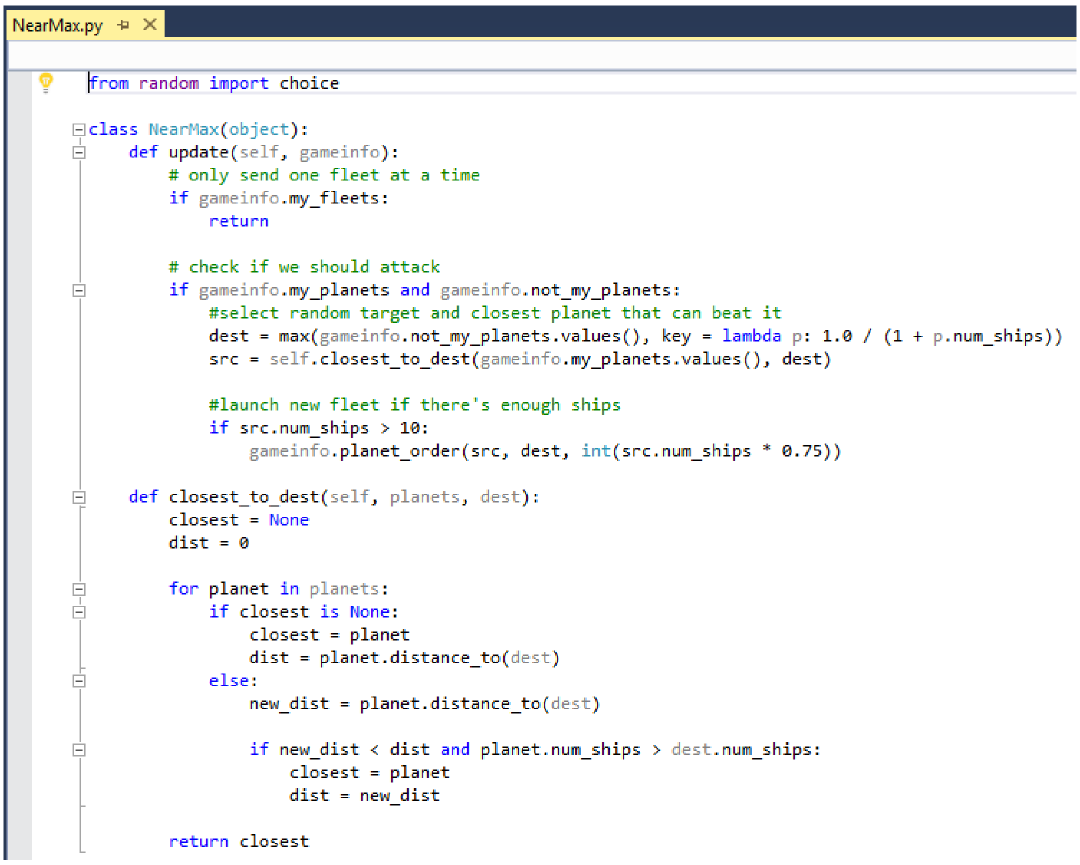
```
Update():
    If there's already a fleet out:
        Do nothing

    If I have planet(s) and there are planets I don't control:
        Dest = planet I don't control with the fewest ships
        Src = ClosestToDest(planets I don't control, Dest)

        If Src has more than 10 ships:
            Send 75% of ships from Src to Dest

ClosestToDest(planets, Dest):
    For each planet:
        If no planet has been selected:
            Closest = planet
            Distance = planet.distanceTo(Dest)
        Else:
            newDist = planet.distanceTo(Dest)

            if newDist < Dist and no. ships on planet > no. ships on Dest:
                Closest = planet
                Dist = newDist

    Return Closest
```


NearMin.py

```python
from random import choice

class NearMin(object):
    def update(self, gameinfo):
        # only send one fleet at a time
        if gameinfo.my_fleets:
            return

        # check if we should attack
        if gameinfo.my_planets and gameinfo.not_my_planets:
            #select random target and closest planet that can beat it
            dest = min(gameinfo.not_my_planets.values(), key = lambda p: p.num_ships)
            src = self.closest_to_dest(gameinfo.my_planets.values(), dest)

            #launch new fleet if there's enough ships
            if src.num_ships > 10:
                gameinfo.planet_order(src, dest, int(src.num_ships * 0.75))

    def closest_to_dest(self, planets, dest):
        closest = None
        dist = 0

        for planet in planets:
            if closest is None:
                closest = planet
                dist = planet.distance_to(dest)
            else:
                new_dist = planet.distance_to(dest)

                if new_dist < dist and planet.num_ships > dest.num_ships:
                    closest = planet
                    dist = new_dist

        return closest
```

NearMax

```
Update():
    If there's already a fleet out:
        Do nothing

    If I have planet(s) and there are planets I don't control:
        Dest = planet I don't control with the most ships
        Src = ClosestToDest(planets I don't control, Dest)

        If Src has more than 10 ships:
            Send 75% of ships from Src to Dest

ClosestToDest(planets, Dest):
    For each planet:
        If no planet has been selected:
            Closest = planet
            Distance = planet.distanceTo(Dest)
        Else:
            newDist = planet.distanceTo(Dest)

            if newDist < Dist and no. ships on planet > no. ships on Dest:
                Closest = planet
                Dist = newDist

    Return Closest
```



```python
from random import choice

class NearMax(object):
    def update(self, gameinfo):
        # only send one fleet at a time
        if gameinfo.my_fleets:
            return

        # check if we should attack
        if gameinfo.my_planets and gameinfo.not_my_planets:
            #select random target and closest planet that can beat it
            dest = max(gameinfo.not_my_planets.values(), key = lambda p: 1.0 / (1 + p.num_ships))
            src = self.closest_to_dest(gameinfo.my_planets.values(), dest)

            #launch new fleet if there's enough ships
            if src.num_ships > 10:
                gameinfo.planet_order(src, dest, int(src.num_ships * 0.75))

    def closest_to_dest(self, planets, dest):
        closest = None
        dist = 0

        for planet in planets:
            if closest is None:
                closest = planet
                dist = planet.distance_to(dest)
            else:
                new_dist = planet.distance_to(dest)

                if new_dist < dist and planet.num_ships > dest.num_ships:
                    closest = planet
                    dist = new_dist

        return closest
```