In [3]:
```
import turtle
import time
```

In [126…
```
def draw_rectangle(t,x,y,height, width, color):
    t.setpos(x,y)
    t.speed(1000)
    t.pendown()
    t.color(color)
    t.begin_fill()
    t.forward(width)
    t.right(90)
    t.forward(height)
    t.right(90)
    t.forward(width)
    t.right(90)
    t.forward(height)
    t.right(90)
    t.end_fill()
    t.penup()
    t.hideturtle()

def draw_stars(t,x,y,color,length) :
    t.goto(x,y)
    t.speed(1000)
    t.setheading(0)
    t.pendown()
    t.begin_fill()
    t.color(color)
    for turn in range(0,5) :
        t.forward(length)
        t.right(144)
        t.forward(length)
        t.right(144)
    t.end_fill()

    t.penup()
    t.hideturtle()

# this function create navy color square

def draw_square(t,x,y,flag_ht):
    square_ht = (7/13) * flag_ht
    square_wdt = (0.76) * flag_ht
    draw_rectangle(t,x,y,square_ht, square_wdt,"navy")
#defining a function for drawing a 6 row star

def stars1(t,ht,sz):
    dist_of_stars = 30
    dist_bet_lines = ht + 6
    y = 112
    for row in range(0,5) :
        x = -234
        for star in range (0,6) :
            draw_stars(t,x, y, "white", sz)
            x = x + dist_of_stars
        y = y - dist_bet_lines
def stars5(t,ht,sz):
    dist_of_stars = 30
    dist_bet_lines = ht + 6
    y = 100
    for row in range(0,4) :
        x = -217
        for star in range (0,5) :
            draw_stars(t,x, y, "white", sz)
            x = x + dist_of_stars
        y = y - dist_bet_lines
```

In [127…

```
def main():
    turtle.TurtleScreen._RUNNING=True
    t = turtle.Turtle()
    x = -250
    y = 120
    flag_ht = 250
    flag_wdt = 475
    stripe_ht = flag_ht/13
    stripe_wdt = flag_wdt
    star_size = 12

    t= turtle.Turtle()
    scr = turtle.getscreen()
    scr.title("Flag of America")
    scr.bgcolor("white")




    # draw stripes
    for stripe in range(0,6):
        for color in ["red", "white"]:
            draw_rectangle(t,x,y,stripe_ht, stripe_wdt, color)
            y = y - stripe_ht
    # create last red stripe
    draw_rectangle(t,x,y,stripe_ht, stripe_wdt, "red")


    # draw navy square
    draw_square(t,-250,120,250)


    # draw stars
    stars1(t,stripe_ht,star_size)
    stars5(t,stripe_ht,star_size)


    t.hideturtle()
    scr.mainloop()
```

In [129…

```
if __name__ == "__main__":
    main()
```

```
---------------------------------------------------------------------------
Terminator                                Traceback (most recent call last)
/var/folders/xz/bjl5b8390kndnwrplxdp8b980000gp/T/ipykernel_19603/3832242952.py
in <module>
      1 if __name__ == "__main__":
----> 2     main()

/var/folders/xz/bjl5b8390kndnwrplxdp8b980000gp/T/ipykernel_19603/870302920.py
 in main()
     21     for stripe in range(0,6):
     22         for color in ["red", "white"]:
---> 23             draw_rectangle(t,x,y,stripe_ht, stripe_wdt, color)
     24             y = y - stripe_ht
     25     # create last red stripe

/var/folders/xz/bjl5b8390kndnwrplxdp8b980000gp/T/ipykernel_19603/3959704425.py
in draw_rectangle(t, x, y, height, width, color)
      6     t.begin_fill()
      7     t.forward(width)
----> 8     t.right(90)
      9     t.forward(height)
     10     t.right(90)

~/opt/anaconda3/lib/python3.9/turtle.py in right(self, angle)
   1677         337.0
   1678         """
-> 1679         self._rotate(-angle)
   1680
   1681     def left(self, angle):

~/opt/anaconda3/lib/python3.9/turtle.py in _rotate(self, angle)
   3278                 self._update()
   3279         self._orient = neworient
-> 3280         self._update()
   3281
   3282     def _newLine(self, usePos=True):

~/opt/anaconda3/lib/python3.9/turtle.py in _update(self)
   2659             return
   2660         elif screen._tracing == 1:
-> 2661             self._update_data()
   2662             self._drawturtle()
   2663             screen._update()                  # TurtleScreenBase

~/opt/anaconda3/lib/python3.9/turtle.py in _update_data(self)
   2645
   2646     def _update_data(self):
-> 2647         self.screen._incrementudc()
   2648         if self.screen._updatecounter != 0:
   2649             return

~/opt/anaconda3/lib/python3.9/turtle.py in _incrementudc(self)
   1291         if not TurtleScreen._RUNNING:
   1292             TurtleScreen._RUNNING = True
-> 1293             raise Terminator
   1294         if self._tracing > 0:
   1295             self._updatecounter += 1

Terminator:
```

In [ ]: