Page 281 Question 13 a

gallons = (0.02 * 1000 * 5)/50

In [52]:
```
gpm = (0.02 * 1000 * 5)/50
```

In [53]:
```
print("gallons per minute = " + str(gpm))
```

```
gallons per minute = 2.0
```

Current water in tank is 1000 gallons. 1010 gallons - 1000 gallons = 10 gallons. Solving for when the tank will have an additional 10 gallons.

In [54]:
```
requiredGallons = 10
```

In [55]:
```
print("The tank will hold 1010 gallons of salt water after " + \
      str(requiredGallons / gpm) + " minutes.")
```

```
The tank will hold 1010 gallons of salt water after 5.0 minutes.
```

In [59]:
```
def TaylorFour(xi,h,t):
    yp = (0.1 - (3.0 * xi)/(1000.0 + 2.0 * t))
    ypp = (15.0 * xi - 0.6 * t - 300)/((1000 + 2.0 * t)**2)
    yppp = (-105.0 * xi + 4.2 * t + 2100)/((1000.0  + 2.0 * t)**3)
    ypppp = (945.0 * xi + 37.8 * t + 13230)/((1000.0  + 2.0 * t)**4)

    result = xi + h*(yp + (h/2.0)*ypp + ((h**2.0)/6.0)*yppp + ((h**3.0)/24.0)*ypppp)

    return result
```

In [60]:
```
TaylorFour(50,0.5,5)
```

Out[60]: 49.97579728518067

Page 292 Question 27

In [45]:
```
def RungeKuttaOF(a, b, n, alpha):
    h = (b - a)/n
    t = a
    w = alpha
    for i in range(1, n + 1):
        k1 = h * f(t,w)
        k2 = h * f((t + h/2.0), (w + k1/2.0))
        k3 = h * f((t + h/2.0), (w + k2/2.0))
        k4 = h * f((t + h), (w + k3))

        w = w + (k1 + (2.0 * k2) + (2.0 * k3) + k4)/6.0

        t = a + (i * h)
    return (t, w)
```

Do not change. Test run from page 289

In [46]:
```
def f(t,y):
    return y - t**2 + 1
```

In [47]:
```
RungeKuttaOF(0, 2, 10, 0.5)
```

Out[47]:
```
(2.0, 5.305363000692653)
```

In [48]:
```
f(0,0.5)
```

Out[48]:
```
1.5
```

End test run

In [49]:
```
k = 6.22 * 10**(-19)
n1 = 2 * 10**(3)
n2 = 2 * 10**(3)
n3 = 3 * 10**(3)
```

In [50]:
```
def f(t,x):
    return k * (((n1-(x/2.0))**(2.0) * (n2-(x/2.0))**(2.0) * (n3-(3.0*x/4.0))**(3.0)))
```

In [51]:
```
RungeKuttaOF(0, 0.2, 1000, 0)
```

Out[51]:
```
(0.2, 2079.4083752829647)
```