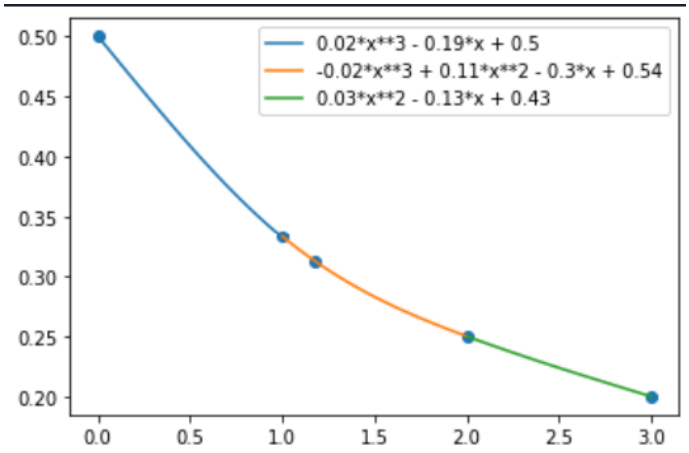
SCL PROBLEM SHEET 7

1.

```
import sympy as sp
import numpy as np
import matplotlib.pyplot as plt
n=int(input("enter number of points: "))
x=[]
y=[]
for i in range(n):
    x.append(float(input("enter x%d : "%(i+1))))
    y.append(float(input("enter f(x%d): "%(i+1))))
xx=float(input("enter x: "))
yy=0
h=x[1]-x[0]
eqns=()
M=[]
for i in range(n):
    if(i==0 or i==n-1):
        M.append(0)
    else:
        M.append(sp.symbols(chr(64+i)))

for i in range(1,n-1):
    s=(6/(h**2))*(y[i-1]-2*y[i]+y[i+1])
    eq=M[i-1]+4*M[i]+M[i+1]-s
    eqns+=(eq,)
M=sp.solve(eqns,tuple(M[1:n-1]))
M=list(M.values())
M=[0]+M+[0]
polys=[]
print("INTERPOLATED POLYNOMIAL: ")
for i in range(n-1):
    xs=sp.symbols('x')
```

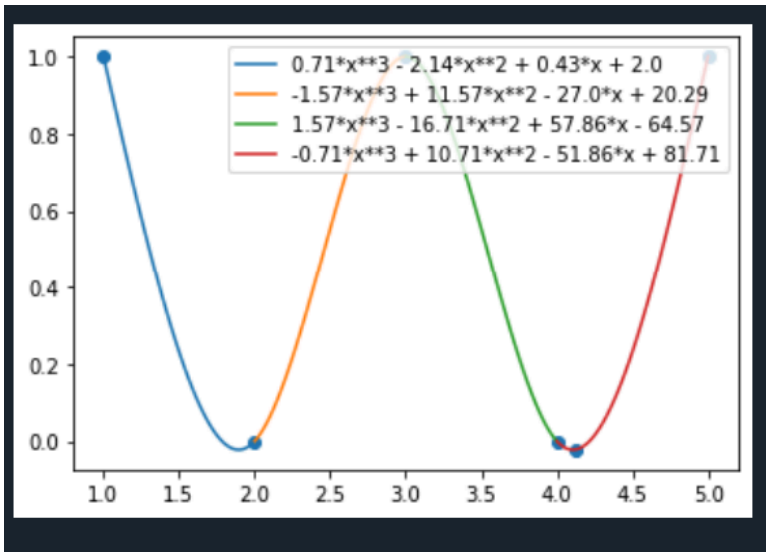
```
fx=((1/6)*((x[i+1]-xs)**3)*M[i])+((1/6)*((xs-x[i])**3)*M[i+1])+((x[i+1]-xs)*(y[i]-(1/6)*M
[i]))+((xs-x[i])*(y[i+1]-(1/6)*M[i+1]))
    fx1=sp.simplify(fx)
    fx1=sp.expand(fx1)
    coeffs = fx1.as_coefficients_dict()
    coeffs = {term: round(float(coeff), 2) for term, coeff in coeffs.items()}
    fx1 = sum(term * coeff for term, coeff in coeffs.items())

    print("%d <= x <= %d"%(x[i],x[i+1]))
    print(fx1)
    fx=sp.lambdify(xs,fx,'numpy')
    xv=np.linspace(x[i],x[i+1])
    plt.plot(xv,fx(xv),label="%s"%fx1)
    if(x[i]<=xx and xx<=x[i+1]):
        yy=fx(xx)
x.append(xx)
y.append(yy)
plt.scatter(x,y)
plt.legend(loc="upper right")
plt.show()
print("f(%f)=%f"%(xx,yy))
```

c)



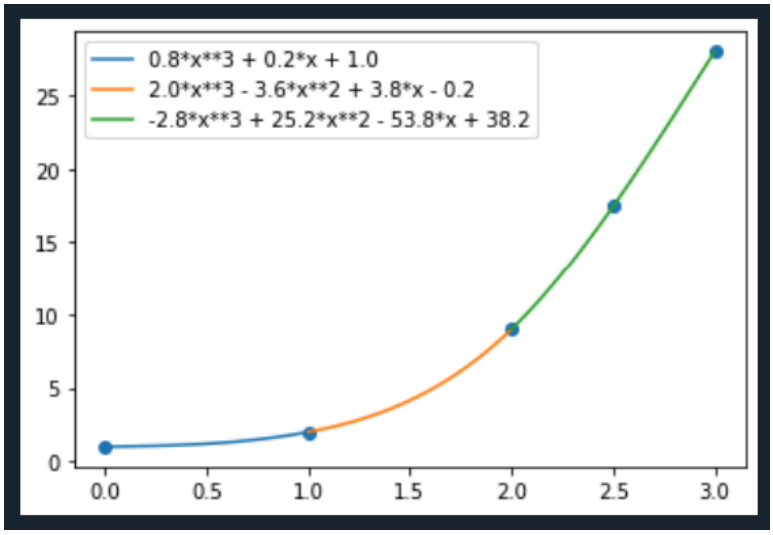

```
enter number of points: 4
enter x1 : 0
enter f(x1): 0.5
enter x2 : 1
enter f(x2): 0.333
enter x3 : 2
enter f(x3): 0.25
enter x4 : 3
enter f(x4): 0.2
enter x: 1.175
[0, 0.121200000000000, 0.0192000000000000, 0]
INTERPOLATED POLYNOMIAL:
0 <= x <= 1
0.02*x**3 - 0.19*x + 0.5
1 <= x <= 2
-0.02*x**3 + 0.11*x**2 - 0.3*x + 0.54
2 <= x <= 3
0.03*x**2 - 0.13*x + 0.43
f(1.175000)=0.312610
```

b)



```
enter number of points: 5
enter x1 : 1
enter f(x1): 1
enter x2 : 2
enter f(x2): 0
enter x3 : 3
enter f(x3): 1
enter x4 : 4
enter f(x4): 0
enter x5 : 5
enter f(x5): 1
enter x: 4.125
[0, 4.28571428571429, -5.14285714285714, 4.28571428571429,
0]
INTERPOLATED POLYNOMIAL:
1 <= x <= 2
0.71*x**3 - 2.14*x**2 + 0.43*x + 2.0
2 <= x <= 3
-1.57*x**3 + 11.57*x**2 - 27.0*x + 20.29
3 <= x <= 4
1.57*x**3 - 16.71*x**2 + 57.86*x - 64.57
4 <= x <= 5
-0.71*x**3 + 10.71*x**2 - 51.86*x + 81.71
f(4.125000)=-0.021484
```

a)



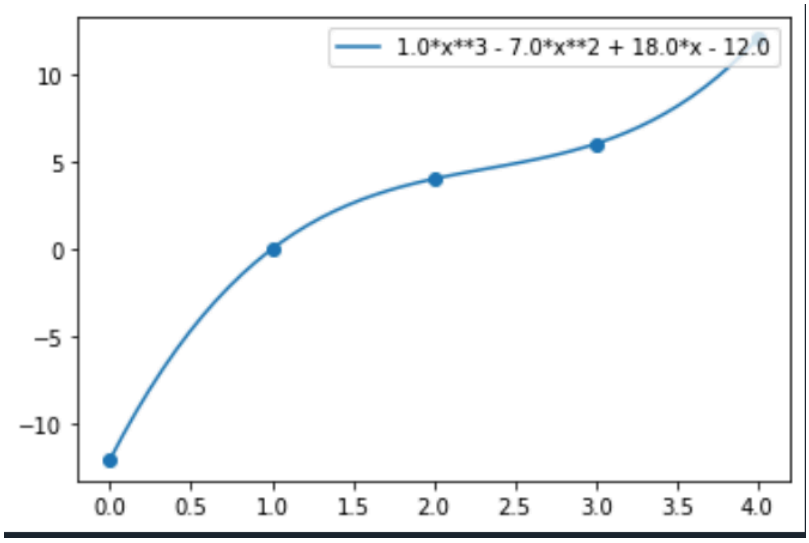

```
enter number of points: 4
enter x1 : 0
enter f(x1): 1
enter x2 : 1
enter f(x2): 2
enter x3 : 2
enter f(x3): 9
enter x4 : 3
enter f(x4): 28
enter x: 2.5
[0, 4.80000000000000, 16.8000000000000, 0]
INTERPOLATED POLYNOMIAL:
0 <= x <= 1
0.8*x**3 + 0.2*x + 1.0
1 <= x <= 2
2.0*x**3 - 3.6*x**2 + 3.8*x - 0.2
2 <= x <= 3
-2.8*x**3 + 25.2*x**2 - 53.8*x + 38.2
f(2.500000)=17.450000
```

2.

```
import sympy as sp
import numpy as np
import matplotlib.pyplot as plt
n=int(input("enter number of points: "))
xl=[]
y=[]
for i in range(n):
    xl.append(float(input("enter x%d : "%(i+1))))
```

```
    y.append(float(input("enter f(x%d): "%(i+1))))
xx=float(input("enter x: "))
yy=0
x=sp.symbols('x')
eq=0
for i in range(0,n):
    f=1
    f1=1
    for j in range(0,n):
        if(i!=j):
            f*=(x-xl[j])
            f1*=(xl[i]-xl[j])
    eq+=(f/f1)*y[i]
eq1=sp.simplify(eq)
eq1=sp.expand(eq1)
coeffs = eq1.as_coefficients_dict()
coeffs = {term: round(float(coeff), 2) for term, coeff in coeffs.items()}
eq1 = sum(term * coeff for term, coeff in coeffs.items())
eq=sp.lambdify(x,eq,'numpy')
print("INTERPOLATED POLYNOMIAL: ")
print(eq1)
yy=eq(xx)
print("f(%f)=%f"%(xx,yy))
xv=np.linspace(xl[0],xl[n-1])
plt.plot(xv,eq(xv),label="%s"%eq1)
xl.append(xx)
y.append(yy)
plt.scatter(xl,y)
plt.legend(loc="upper right")
plt.show()
```

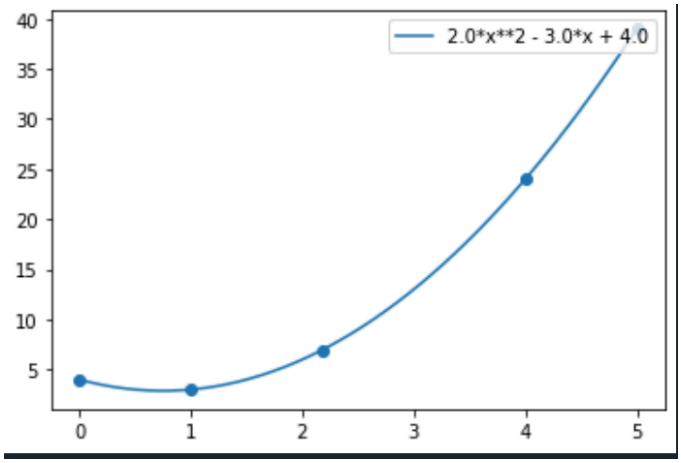
c)



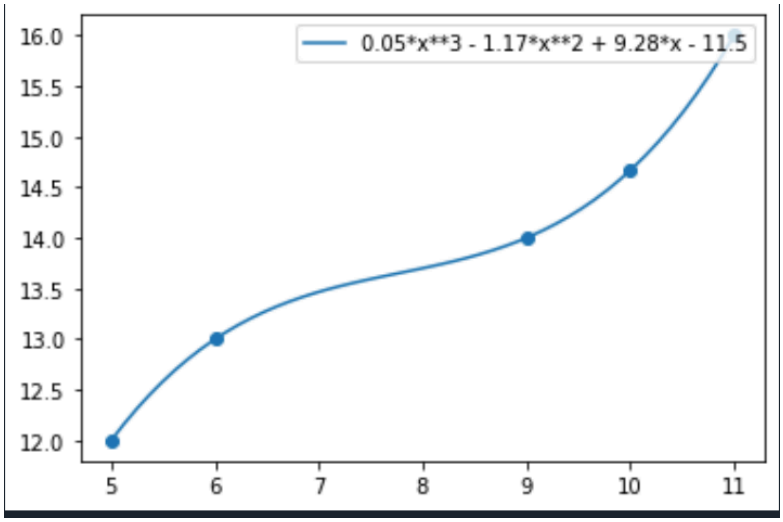

```
enter number of points: 4
enter x1 : 0
enter f(x1): -12
enter x2 : 1
enter f(x2): 0
enter x3 : 3
enter f(x3): 6
enter x4 : 4
enter f(x4): 12
enter x: 2
INTERPOLATED POLYNOMIAL:
1.0*x**3 - 7.0*x**2 + 18.0*x - 12.0
f(2.000000)=4.000000
```

b)

```
enter number of points: 4
enter x1 : 0
enter f(x1): 4
enter x2 : 1
enter f(x2): 3
enter x3 : 4
enter f(x3): 24
enter x4 : 5
enter f(x4): 39
enter x: 2.175
INTERPOLATED POLYNOMIAL:
2.0*x**2 - 3.0*x + 4.0
f(2.175000)=6.936250
```



a)



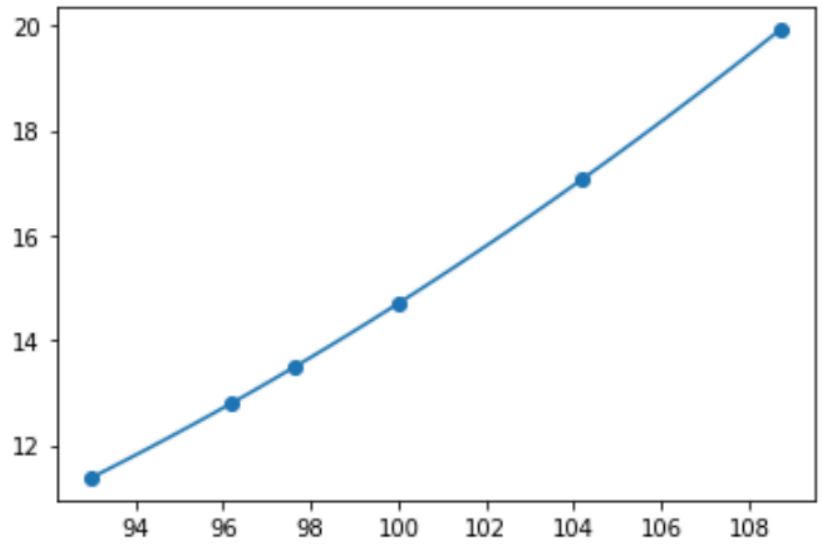

```
enter number of points: 4
enter x1 : 5
enter f(x1): 12
enter x2 : 6
enter f(x2): 13
enter x3 : 9
enter f(x3): 14
enter x4 : 11
enter f(x4): 16
enter x: 10
INTERPOLATED POLYNOMIAL:
0.05*x**3 - 1.17*x**2 + 9.28*x - 11.5
f(10.000000)=14.666667
```

3.

```
import sympy as sp
import numpy as np
import matplotlib.pyplot as plt
n=int(input("enter number of points: "))
xl=[]
yl=[]
for i in range(n):
    xl.append(float(input("enter x%d : "%(i+1))))
    yl.append(float(input("enter f(x%d): "%(i+1))))
yy=float(input("enter y: "))
```

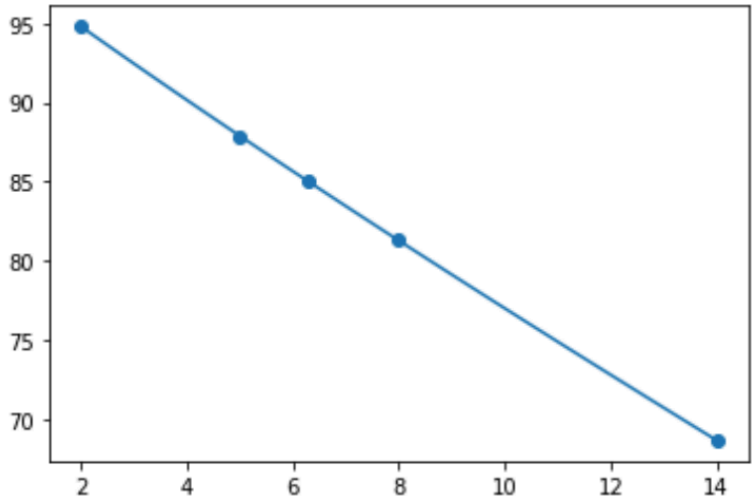
```
xx=0
x=sp.symbols('x')
y=sp.symbols('y')
eq=0
for i in range(0,n):
    f=1
    f1=1
    for j in range(0,n):
        if(i!=j):
            f*=(y-yl[j])
            f1*=(yl[i]-yl[j])
    eq+=(f/f1)*xl[i]
eq1=sp.simplify(eq)
eq1=sp.expand(eq1)
eq=sp.lambdify(y,eq,'numpy')
xx=eq(yy)
print(xx,yy)
print("INTERPOLATED POLYNOMIAL: ")
print(eq1)
print("f^-1(%f)=%0.2f"%(yy,xx))
yv=np.linspace(yl[0],yl[n-1])
plt.plot(eq(yv),yv,label="%s"%eq1)
xl.append(xx)
yl.append(yy)
plt.scatter(xl,yl)
plt.show()
```

b)




```
enter number of points: 5
enter x1 : 93
enter f(x1): 11.38
enter x2 : 96.2
enter f(x2): 12.8
enter x3 : 100
enter f(x3): 14.7
enter x4 : 104.2
enter f(x4): 17.07
enter x5 : 108.7
enter f(x5): 19.91
enter y: 13.5
97.6557503056373 13.5
INTERPOLATED POLYNOMIAL:
-0.000187934989896998*y**4 + 0.0145574185897743*y**3 - 0.452437568334744*y**2 +
8.13566807210373*y + 40.7065418337443
f^-1(13.500000)=97.66
```

a)




```
enter number of points: 4
enter x1 : 2
enter f(x1): 94.8
enter x2 : 5
enter f(x2): 87.9
enter x3 : 8
enter f(x3): 81.3
enter x4 : 14
enter f(x4): 68.7
enter y: 85
6.303830017160332 85.0
INTERPOLATED POLYNOMIAL:
1.28953881244364e-5*y**3 - 0.00194046795745861*y**2 - 0.403241775126169*y +
46.6798816636078
f^-1(85.000000)=6.30
```

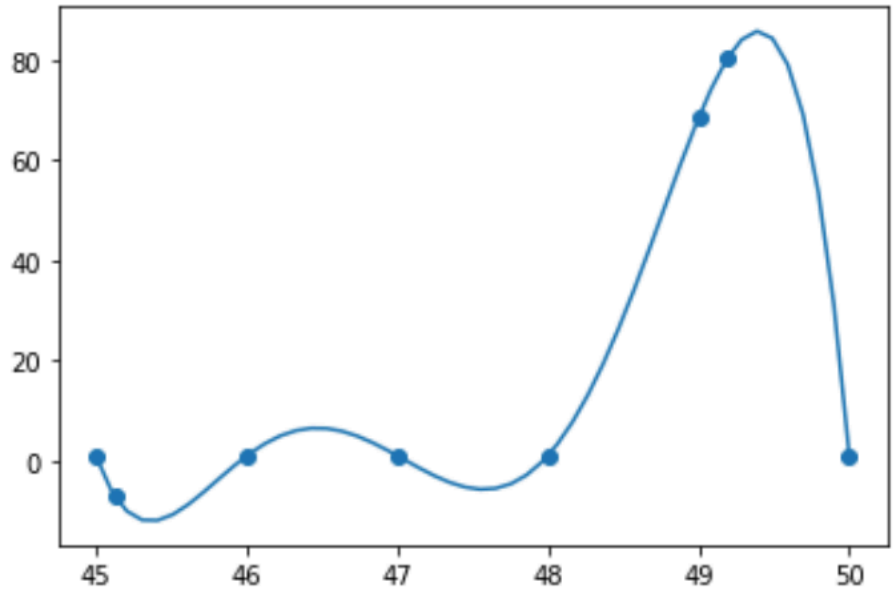
4.

```
import numpy as np
import sympy as sp
import math
import matplotlib.pyplot as plt
n=int(input("enter number of points: "))
x=[]
y=[]
for i in range(n):
    x.append(float(input("enter x%d : "%(i+1))))
    y.append(float(input("enter f(x%d): "%(i+1))))
xx=float(input("enter x: "))
xx1=float(input("enter x: "))
yy=0
yy=0
```

```
# n=6
# x=[45,46,47,48,49,50]
# y=[1.0000,1.03553,1.07237,1.11061,68.48,1.19175]
itable=np.zeros((n,n+1))
for i in range(n):
    itable[i][0]=x[i]
for i in range(0,n):
    itable[i][1]=y[i]
for i in range(2,n+1):
    for j in range(n-(i-1)):
        itable[j][i] = itable[j+1][i-1]-itable[j][i-1]
print("INTERPOLATION TABLE")
print(itable)
np.set_printoptions(suppress=True)
np.round(itable,decimals=2)
xs=sp.symbols('x')
p=(xs-x[0])/(x[1]-x[0])
itable=itable[:,1:n+1]
eq=itable[0][0]
for i in range(1,n):
    k=1
    for j in range(1,i+1):
        k=k*(p-j+1)
    k=k*itable[0][i]
    k=k/math.factorial(i)
    eq+=k
eq1=sp.simplify(eq)
eq=sp.lambdify(xs,eq,'numpy')
print("INTERPOLATED POLYNOMIAL: ")
print(eq1)
print("f(%f)=%0.4f"%(xx,eq(xx)))
print("f(%f)=%0.4f"%(xx1,eq(xx1)))
xv=np.linspace(x[0],x[n-1])
plt.plot(xv,eq(xv),label="%s"%eq1)
x.append(xx)
y.append(eq(xx1))
x.append(xx)
```

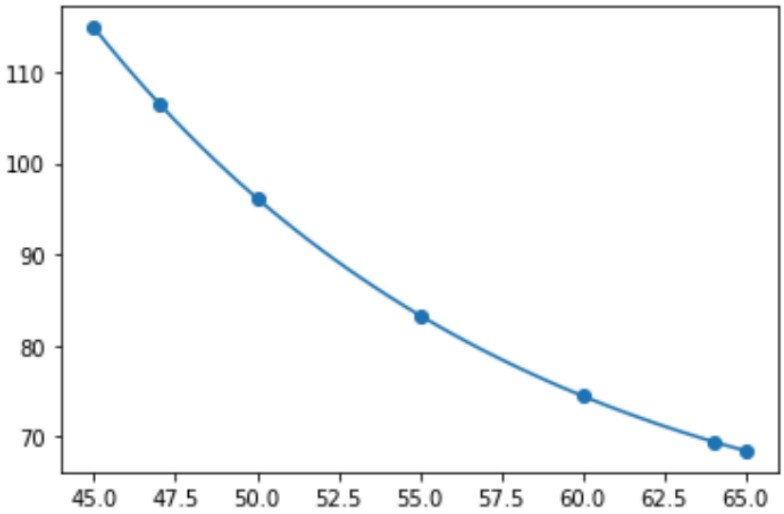
```
y.append(eq(xx1))
plt.scatter(x,y)
plt.show()
```

c)



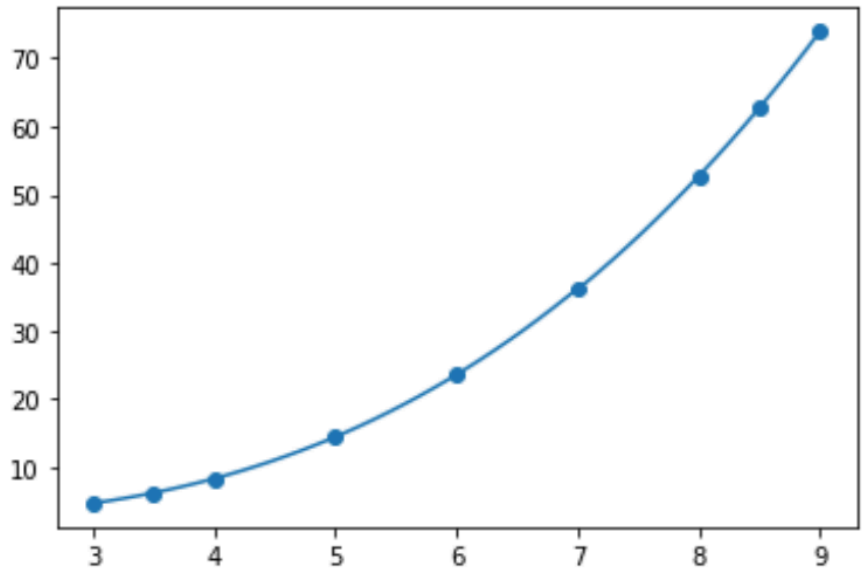

```
INTERPOLATION TABLE
[[  45.         1.         0.03553    0.00131    0.00009   67.32966
  -336.6482 ]
 [  46.         1.03553    0.03684    0.0014    67.32975 -269.31854
     0.     ]
 [  47.         1.07237    0.03824   67.33115 -201.98879    0.
     0.     ]
 [  48.         1.11061   67.36939 -134.65764    0.         0.
     0.     ]
 [  49.        68.48     -67.28825    0.         0.         0.
     0.     ]
 [  50.         1.19175    0.         0.         0.         0.
     0.     ]]
INTERPOLATED POLYNOMIAL:
-2.80540166666667*x**5 + 662.074794166667*x**4 - 62479.1006583333*x**3 + 2947063.23862083*x**2 -
69482000.530525*x + 655050070.74925
f(45.125000)=-7.0592
f(49.189000)=80.2639
```

b)




```
INTERPOLATION TABLE
[[ 45.   114.84 -18.68   5.84  -1.84   0.68]
 [ 50.    96.16 -12.84   4.    -1.16   0.  ]
 [ 55.    83.32  -8.84   2.84   0.     0.  ]
 [ 60.    74.48  -6.     0.     0.     0.  ]
 [ 65.    68.48   0.     0.     0.     0.  ]]
INTERPOLATED POLYNOMIAL:
4.5333333333331e-5*x**4 - 0.0119733333333328*x**3 + 1.23166666666663*x**2 -
59.1126666666652*x + 1185.95999999998
f(47.000000)=106.5212
f(64.000000)=69.4856
```

a)




```
INTERPOLATION TABLE
[[ 3.   4.8  3.6  2.5  0.5 -0.  -0.   0. ]
 [ 4.   8.4  6.1  3.   0.5 -0.   0.   0. ]
 [ 5.  14.5  9.1  3.5  0.5  0.   0.   0. ]
 [ 6.  23.6 12.6  4.   0.5  0.   0.   0. ]
 [ 7.  36.2 16.6  4.5  0.   0.   0.   0. ]
 [ 8.  52.8 21.1  0.   0.   0.   0.   0. ]
 [ 9.  73.9  0.   0.   0.   0.   0.   0. ]]
INTERPOLATED POLYNOMIAL:
4.81096644004235e-17*x**6 - 1.59502041204481e-15*x**5 +
2.14088006581884e-14*x**4 + 0.0833333333331845*x**3 + 0.250000000000566*x**2 -
1.23333333333445*x + 4.00000000000089
f(3.500000)=6.3187
f(8.500000)=62.7563
```