Assignment 4

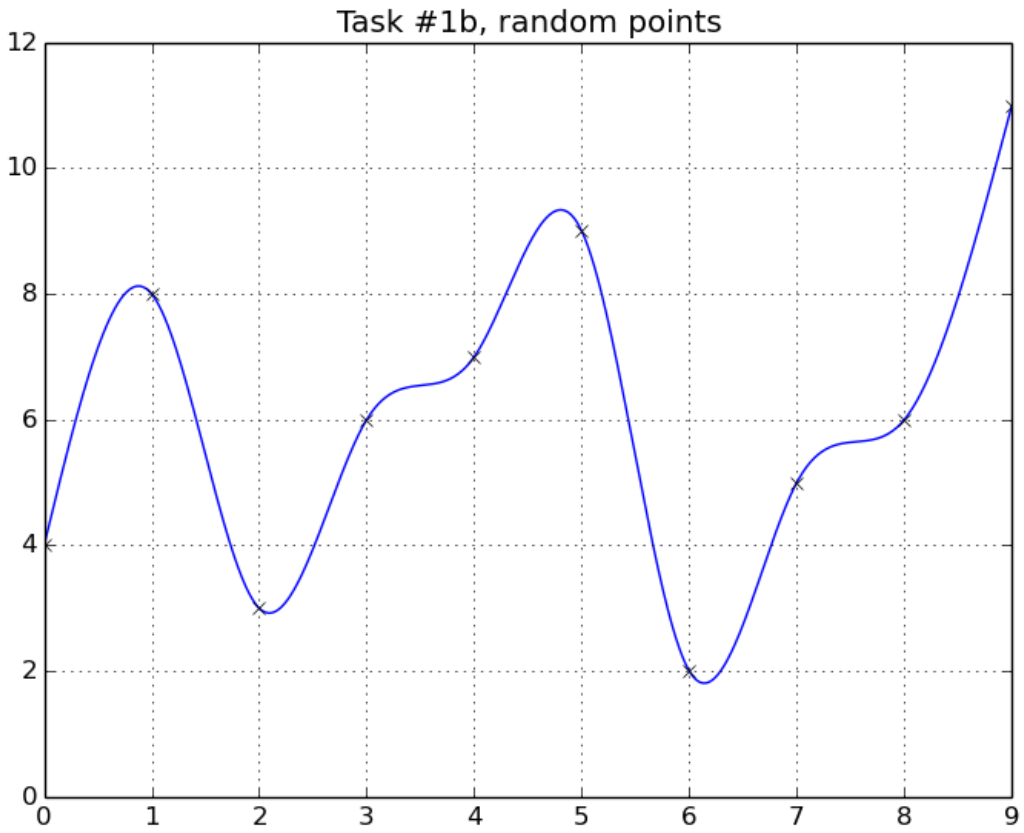
Task #1a

By - Sebastian Elm, Hugo Hjertén, Albin Johansson, Adnan Mehmedagic, Axel Lundholm

```
def cubeSpline(xint,yint):
    h = abs(xint[1]-xint[0])
    di = yint
    m = len(xint) # antal punkter
    coeff = zeros((m-1,4))
    allSigmas = zeros((m,m))
    allSigmas[0][0]=1
    allSigmas[m-1][m-1]=1
    Y = zeros((m,1))
    col=0
    for row in range(1,m-1):
        allSigmas[row][col]=1
        col+=1
        allSigmas[row][col]=4
        col+=1
        allSigmas[row][col]
        col-=1
        Y[row][0] =
(6/(h**2))*(yint[row+1]-2*yint[row]+yint[row-1])
    sigmas = solve(allSigmas,Y)
    bi = sigmas/2
    ai=[]
    ci=[]
    for i in range(m-1):
        ai.append((sigmas[i+1]-sigmas[i])/(6*h))
        ci.append((yint[i+1]-yint[i])/h -
(2*sigmas[i]+sigmas[i+1])*h/6)
    for i in range(m-1):
        coeff[i][0]=ai[i]
        coeff[i][1]=bi[i]
        coeff[i][2]=ci[i]
        coeff[i][3]=di[i]
    return(coeff)
```

The function returns a matrix with all the spline polynomials.

Task #1b

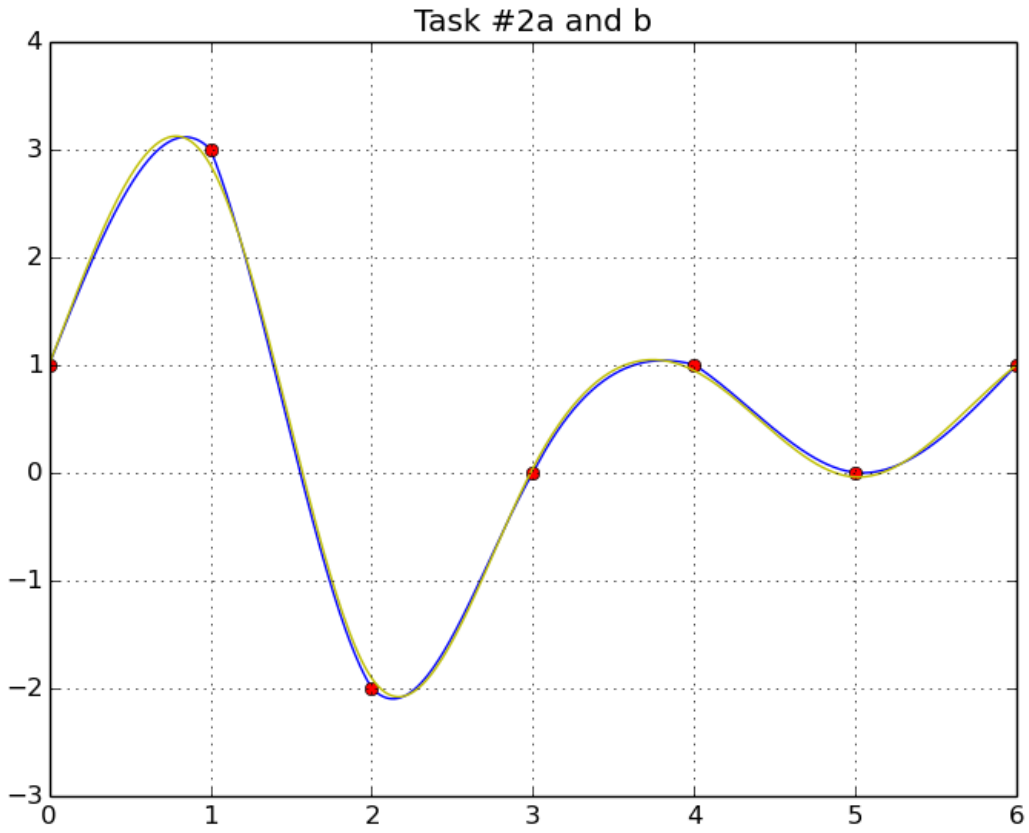


The random points used where the following:

```
x = linspace(0,9,10)
```

```
y = [4, 8, 3, 6, 7, 9, 2, 5, 6, 11]
```

Task #2b



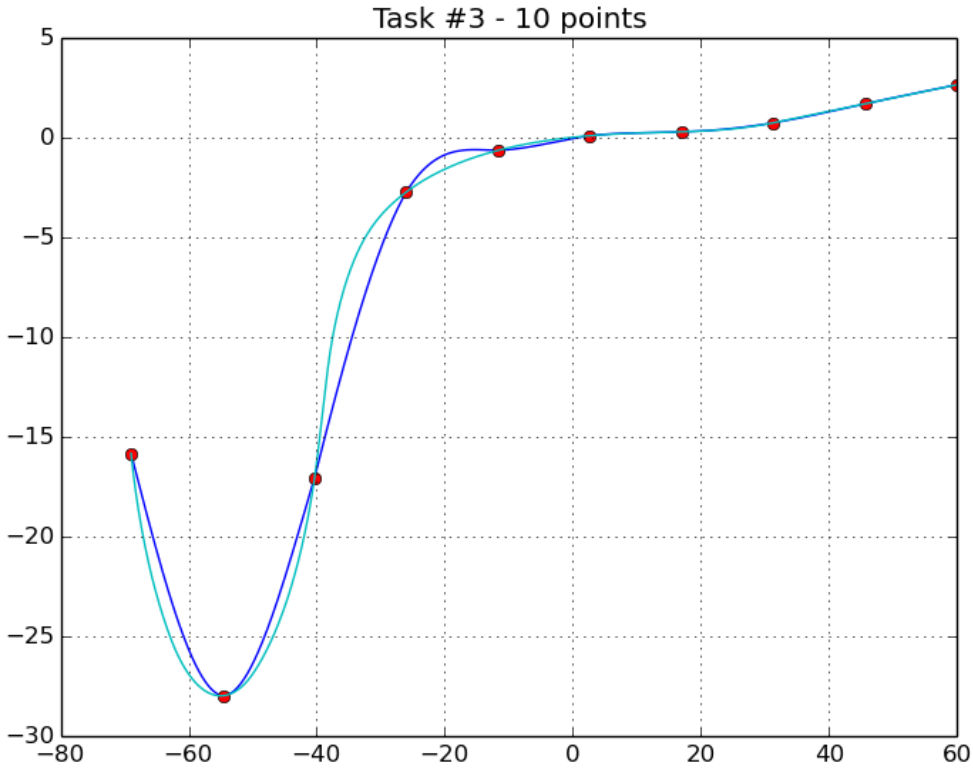
In the plot above we see the cubic spline (the blue curve) that goes through all the points and the cubic B-spline (the yellow curve) that doesn't go through all the points exactly. The x-values, *xi*, and de Boor points, *di*, used that we found to give the most optimal curve were:

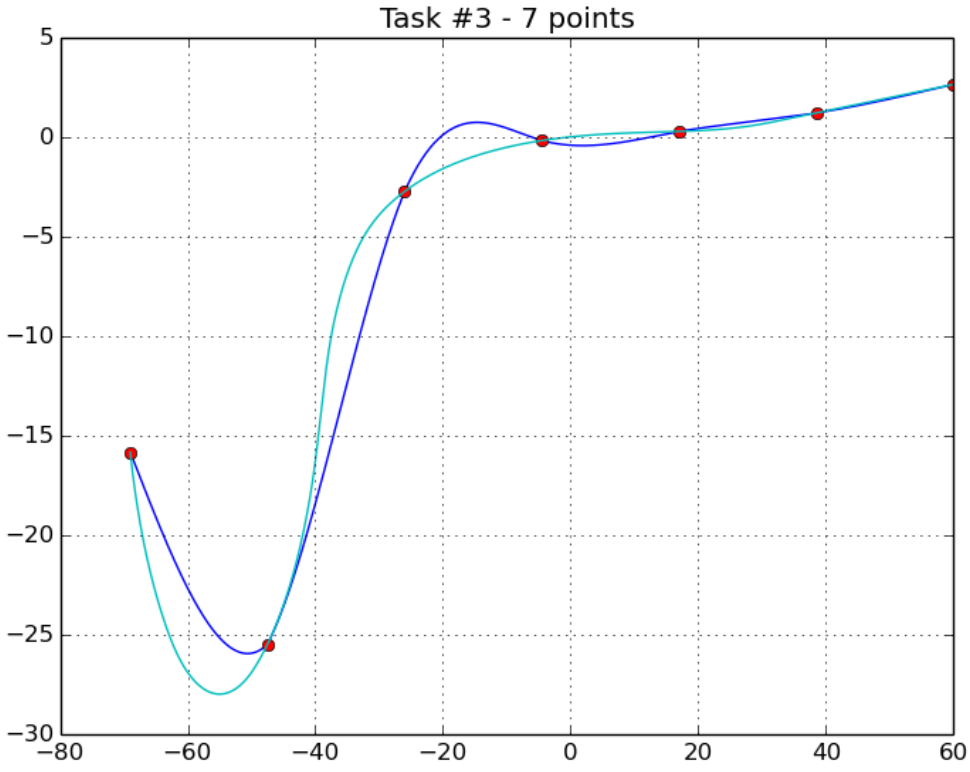
```
xi= [0,0,0,0, 1, 2, 3, 4, 5, 6,6,6,6] # The x-vector with 3 extra nodes on each sides,
value of the endpoints.

di= [1.,2.3,5.15,-4.35,0.8,1.4,-0.73,0.6,1.] # De Boor "control points" to the B-spline
function two extra on each side.
```

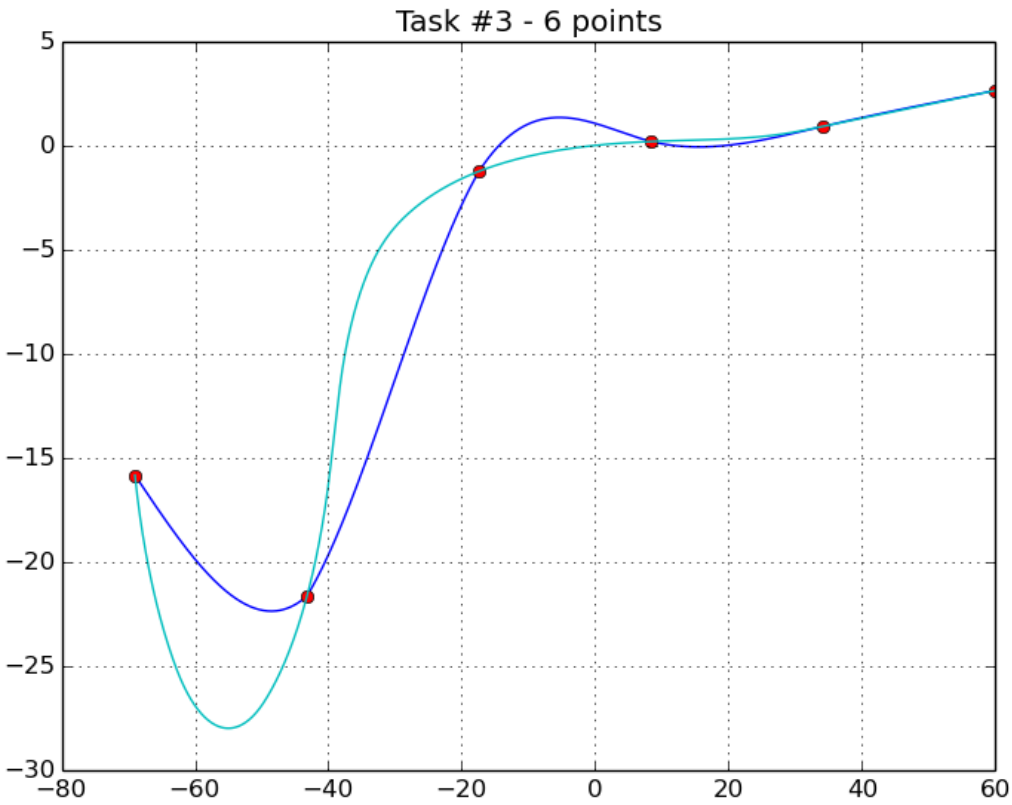
Hugos explanation (we might want to write an individual explanation each… hehe)
To summarize: 2 extra de Boor points added (2 extra B-splines), 1 on each side of interval. This means 6 extra points (xi in the assignment), 3 on both sides. This is because 1) one point missing to build the B-spline for the second point and the second-last point 2) two points missing to build the B-spline for the first and last point 3) three points missing to build the extra B-spline on each side of the interval. However, out of these three B-splines mentioned above, they share many points. In total, three extra points are needed on each side.

The optimal value of n should therefore be 9, since we add 2 more splines to the original 7. (m+3, where m is 6, max index).

Task #3


Task #3 - 10 points
5
0
−5
−10
−15
−20
−25
−30
−80
−60
−40
−20
0
20
40
60


Task #3 - 7 points
5
0
−5
−10
−15
−20
−25
−30
−80
−60
−40
−20
0
20
40
60



We plotted the wheel form using our cubic spline function from Task #1, using a different amount of points. Tadaa.