In [1]:
```
import numpy as np
import numpy.linalg as la
import matplotlib.pyplot as plt
import matplotlib.lines as lns

def plot_tangent(point,k):
    limits=plt.axis()
    x=np.linspace(limits[0],limits[1],100)
    plt.plot(x,point[1]+k*(x-point[0]), linewidth=5)


def plot_function(interval,fun):
    a=interval[0]
    b=interval[1]

    x=np.linspace(a-1,b+1,100)
    y1=fun(x)

    plt.figure(figsize=(15, 10))
    plt.plot(x,y1,linewidth=5)


def calculate_error(x0,fun,min_st_size,method,solution):
    step = 1
    errors=[]
    sub_intervals=[]

    while step >= min_st_size :
        y = method(x0,step,fun)
        errors.append(np.abs(solution-y))
        sub_intervals.append(step)
        step = step / 2

    plt.plot(np.arange(0,len(errors)),errors,linewidth=5)
    plt.xlabel('Indeks i gresaka, za koji vazi: h=1/2^{i-1}. Npr. za i=1 je h=1, za i=2 je h=1/2 itd.',
fontsize=18)
    plt.ylabel('Greska',fontsize=18)

    return [errors,sub_intervals]
```

Numeričko diferenciranje

Cilj nam je da izvod u tački odredimo numerički, a ne simbolički.

Numeričke metode za određivanje izvoda mogu da se koriste kada je izvod funkcije teško odrediti analitički ili kada nemamo simbolički oblik funkcije već su nam samo date njene tačke.

Na primer, možemo da imamo podatke od pređenom putu po vremenu za neki objekat, a cilj nam je da odredimo njegovu brzinu i ubrzanje.

Recimo da imamo primer objekta koji je u slobodnom padu koji smo obrađivali ranije. Pokušaćemo da krenemo od pređenog puta i da otkrijemo brzinu i ubrzanje.

Znamo da je prvi izvod promene položaja tela u stvari brzina tela. Cilj nam je odredimo prvi izvod na osnovu tabelarnih podataka.

In [2]:
```python
import numpy as np

def calculate_force(m,c,v):
#g (gravitational constant) = 9.8 m/s² (in the negative y direction)
#m (mass of body)
#C (drag constant)
#v (velocity of body)
    g = 9.8
    Fd = c * v ** 2 #F drag
    Fg = m * g #F gravity
    F = Fg - Fd
    return F

def simulate(m,c,v0,h0,t0,t1,step):
    time = np.arange(t0, t1 + step, step)
    n = len(time)

    velocity = np.zeros(n)
    height = np.zeros(n)

    height[0] = h0
    velocity[0] = v0

    for i in range(1,n):
        F = calculate_force(m,c,velocity[i-1])

        velocity[i] = velocity[i-1] + step * F / m

        height[i] = height[i-1] - step * velocity[i] #znak minus je jer telo pada pa mu se visina smanj
uje

    return [time,velocity,height]
```

Uzećemo promenu položaja u prvih 10 koraka. Vrednost koeficijenta trenja namerno je postavljena na 0 da bi kasnije u toku predavanja pokazali kako izgleda ubrzanje.

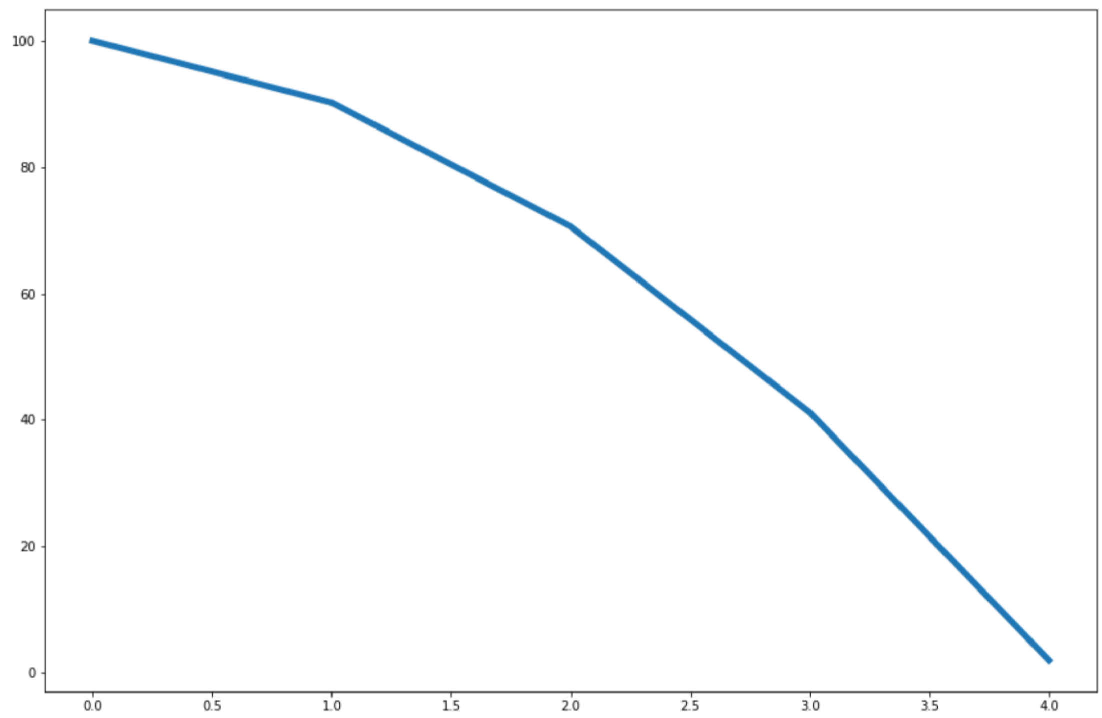
In [9]:
```python
m = 1; #kg
c = 0; #kg/m
h0 = 100 #m
v0 = 0; #m/s
step = 1; #s
t0 = 0
t1 = 4

[time,velocity,height]=simulate(m,c,v0,h0,t0,t1,step)
print(time)
print(velocity)
print(height)
```

```
[0 1 2 3 4]
[ 0.   9.8 19.6 29.4 39.2]
[100.   90.2  70.6  41.2   2. ]
```

Crtamo grafik promene položaja

In [10]:
```python
plt.rcParams['figure.figsize']=(15, 10)

plt.plot(time,height,linewidth=5)
```

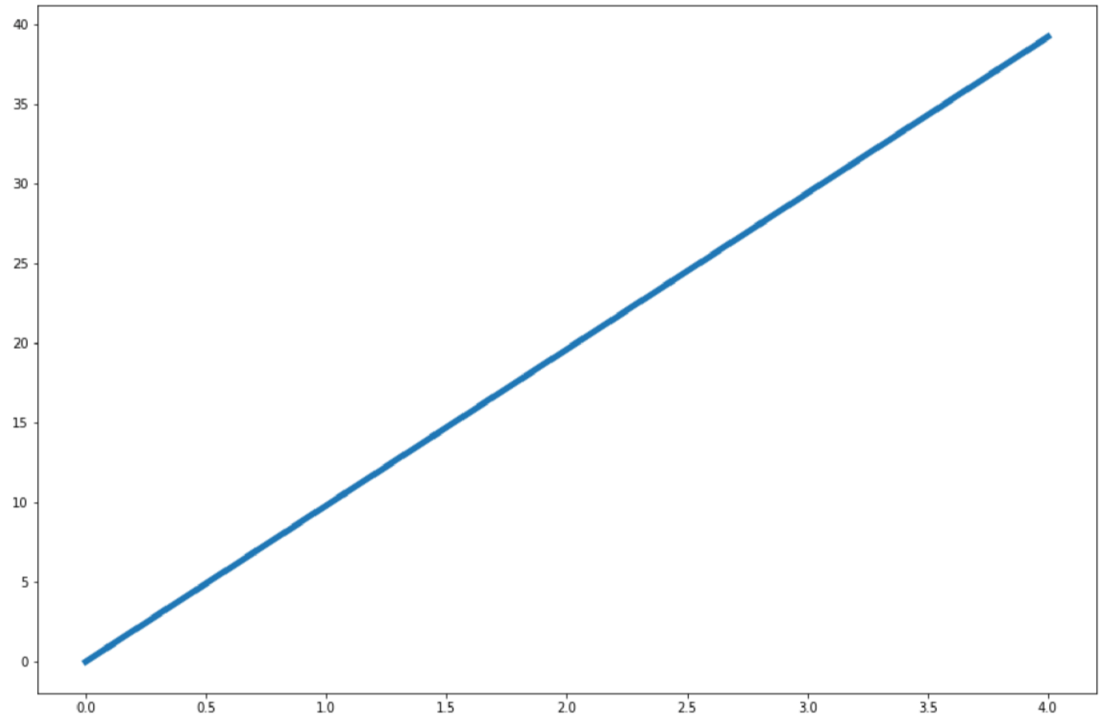
Out[10]:
```
[<matplotlib.lines.Line2D at 0x189c9d4f588>]
```



Crtamo grafik promene brzine

In [11]:
```
plt.plot(time,velocity,linewidth=5)
```

Out[11]:
```
[<matplotlib.lines.Line2D at 0x189c9daf630>]
```



Vrednosti promene položaja su sledeće:

$t = (0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9)$

$visina = (100.0, 99.9, 99.7, 99.4, 99.0, 98.5, 97.9, 97.2, 96.5, 95.6)$

Kod koji je dat u nastavku prolazi kroz niz $visina$ i računa tzv. konačne razlike, konkretno $razliku\ unapred$:

$$brzina_i = \frac{visina_{i+1} - visina_i}{\Delta t}$$

gde je Δt vremenski korak koji je u ovom slučaju 0.1. Razlika unapred je numerička aproksimacija prvog izvoda.

In [12]:
```
n=len(height)
step=1
diff=np.zeros(n-1)

for i in range(n-1):

    h_now=height[i]

    h_next=height[i+1]

    diff[i]=( h_next - h_now ) / step

print(height)
print(diff)
print(velocity)
```

```
[100.   90.2  70.6  41.2   2. ]
[ -9.8 -19.6 -29.4 -39.2]
[ 0.   9.8 19.6 29.4 39.2]
```

Vidimo da se naša aproksimacija poklapa sa stvarnim vrednostima brzine. (Vrednosti liste $diff$ su negativne jer se u ovom konkretnom primeru visina smanjuje.)

Konačne razlike

Videli smo da, pomoću konačnih razlika, možemo uspešno da aproksimiramo izvod iz tabelarnih podataka. U nastavku se detaljno bavimo konačnim razlikama.

Podsetimo se prvo na koji način se definiše prvi izvod funkcije:

$$\frac{df}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

Rešavanjem granične vrednosti dobija se prvi izvod u simboličkom obliku.

Pomoću numeričkog diferenciranja ne možemo da odredimo prvi izvod u simboličkom obliku, ali možemo da odredimo vrednost prvog izvoda u proizvoljnoj tački x_0:

$$f'(x_0) = ?$$

Na primer, numeričko diferenciranje može da reši sledeći problem:

$$f'(2^x) = ?$$
$$x = 0.2$$

Tejlorov red

Osnova za sve formule za numeričko diferenciranje biće nam Tejlorov red. Ako uzmemo da je $x = x_0 + h$, Tejlorov red ima sledeći oblik:

$$f(x_0 + h) = f(x_0) + f'(x_0)h + \frac{1}{2}f''(x_0)h^2 + \frac{1}{3!}f^{(3)}(x_0)h^3 + \ldots$$

Konačne razlike za prvi izvod

Razlika unapred

Kod razlike unapred da bi odredili izvod u nekoj tački x_0 koristimo tačku $x_0 + h$, gde je h proizvoljno odabrani korak.

Razliku unapred izvodimo na sledeći način:

$$f(x_0 + h) = f(x_0) + f'(x_0)h + \frac{1}{2}f''(x_0)h^2$$

$$f(x_0 + h) - f(x_0) = f'(x_0)h + \frac{1}{2}f''(x_0)h^2$$

$$f'(x_0) = \frac{f(x_0 + h) - f(x_0)}{h} - \frac{1}{2}f''(x_0)h$$

Što nam daje formulu:

$$f'(x_0) = \frac{f(x_0+h) - f(x_0)}{h} - O(h)$$

Izvešćemo sada još dve formule, pa ćemo onda ih onda sve zajdeno dalje obraditi.

Razlika unazad

Kod razlike unazad da bi odredili izvod u nekoj tački x_0 koristimo tačku $x_0 - h$, gde je h proizvoljno odabrani korak.

Razliku unazad izvodimo na sledeći način:

$$f(x_0 - h) = f(x_0) - f'(x_0)h + \frac{1}{2}f''(x_0)h^2$$

$$f(x_0 - h) - f(x_0) = -f'(x_0)h + \frac{1}{2}f''(x_0)h^2$$

$$f'(x_0) = \frac{f(x_0) - f(x_0-h)}{h} + \frac{1}{2}f''(x_0)h$$

Što nam daje formulu:

$$f'(x_0) = \frac{f(x_0) - f(x_0-h)}{h} + O(h)$$

Centralna razlika

Kod centralne razlike da bi odredili izvod u nekoj tački x_0 koristimo tačke $x_0 - h$ i $x_0 - h$, gde je h proizvoljno odabrani korak.

Centralnu razliku izvodimo na sledeći način:

$$f(x_0 + h) = f(x_0) + f'(x_0)h + \frac{1}{2}f''(x_0)h^2 + \frac{1}{6}f^{(3)}(x_0)h^3$$

$$f(x_0 - h) = f(x_0) - f'(x_0)h + \frac{1}{2}f''(x_0)h^2 - \frac{1}{6}f^{(3)}(x_0)h^3$$

Oduzimamo predhodna dva Tejlorova razvoja:

$$f(x_0 + h) - f(x_0 - h) = 2f'(x_0)h + 2\frac{1}{6}f^{(3)}(x_0)h^3$$

$$f'(x_0) = \frac{f(x_0+h) - f(x_0-h)}{2h} - frac13f^{(3)}(x_0)h^2$$

Što nam daje formulu:

$$f'(x_0) = \frac{f(x_0+h) - f(x_0-h)}{2h} - O(h^2)$$

Primenićemo sada sve tri formule na primer određivanja izvoda funkcije:

$$f'(2^x) = ?$$
$$x = 2$$

Analitičko rešenje je:

$$f'(2^x) = 2^x \cdot ln(2)$$

$$f'(2) = 2^2 \cdot ln(2) = 2.7726$$

Dakle, imamo da je: $f(x) = 2^x$,$x_0 = 2$ **i** $h = 0.5$. **Za primenu numeričkih metoda biramo prvo** $h = 0.5$, **pa onda** $h = 0.25$.

Razlika unapred

$$f'(x_0) = \frac{f(x_0+h) - f(x_0)}{h} = \frac{2^{2+0.5} - 2^2}{0.5} = 3.3137$$

Relativna (procentualna) greška je:

$$E_R = \left|\frac{2.7726 - 3.3137}{2.7726}\right| = 0.1952 = 19.52\%$$

Razlika unazad

$$f'(x_0) = \frac{f(x_0) - f(x_0-h)}{h} = \frac{2^2 - 2^{2-0.5}}{0.5} = 2.2431$$

$$E_R = \left|\frac{2.7726 - 2.2431}{2.7726}\right| = 0.1549 = 15.49\%$$

Centralna razlika

$$f'(x_0) = \frac{f(x_0+h) - f(x_0-h)}{2h} = \frac{2^{2+0.5} - 2^{2-0.5}}{2 \cdot 0.5} = 2.8284$$

$$E_R = \left|\frac{2.7726 - 2.8284}{2.7726}\right| = 0.0201 = 2.01\%$$

Koristimo sada $h = 0.25$.

Razlika unapred

$$f'(x_0) = \frac{f(x_0+h) - f(x_0)}{h} = \frac{2^{2+0.25} - 2^2}{0.25} = 3.0273$$

Relativna (procentualna) greška je:

$$E_R = \left|\frac{2.7726 - 3.0273}{2.7726}\right| = 0.091 = 9.1\%$$

Razlika unazad

$$f'(x_0) = \frac{f(x_0) - f(x_0-h)}{h} = \frac{2^2 - 2^{2-0.25}}{0.25} = 2.5457$$

$$E_R = \left|\frac{2.7726 - 2.5457}{2.7726}\right| = 0.0818 = 8.18\%$$

Centralna razlika

$$f'(x_0) = \frac{f(x_0+h) - f(x_0-h)}{2h} = \frac{2^{2+0.25} - 2^{2-0.25}}{2 \cdot 0.25} = 2.7865$$

$$E_R = \left|\frac{2.7726 - 2.7865}{2.7726}\right| = 0.005 = 0.5\%$$

Šta možemo da zaključimo iz dobijenih rezultata?

Pre svega, razlika unapred i unazad imaju sličnu grešku dok je greška centralne razlike manja.

Pored toga, vidi se da razlika unapred i unazad imaju red greške $O(h)$ **- prepolovili smo** h, **i greška se takođe prepolovila. Dok je red greške centralne razlike** $O(h^2)$ **- prepolovili smo** h, **a greška se smanjila četiri puta.**

Pokazaćemo sada pomoću grafika kako izgledaju konačne razlike u odnosu na tačnu vrednost izvoda (koeficijent pravca tangente). Koristimo primer koji smo upravo uradili. Pored toga pišemo i kod za konačne razlike, koji je veoma jednostavan.

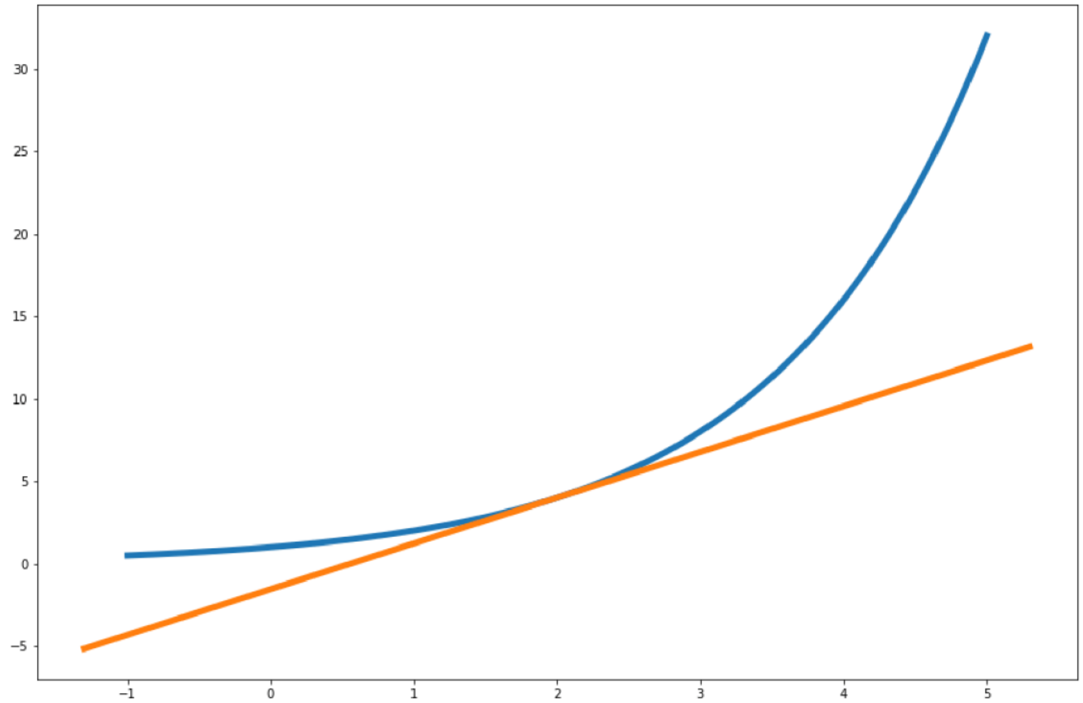
In [13]:
```
def razlika_unapred(x,h,funkcija):
    return ( funkcija(x+h) - funkcija(x) ) / h
```

In [14]:
```
def razlika_unazad(x,h,funkcija):
    return ( funkcija(x) - funkcija(x-h) ) / h
```

In [15]:
```
def centralna_razlika(x,h,funkcija):
    return ( funkcija(x+h) - funkcija(x-h) ) /(2*h)
```

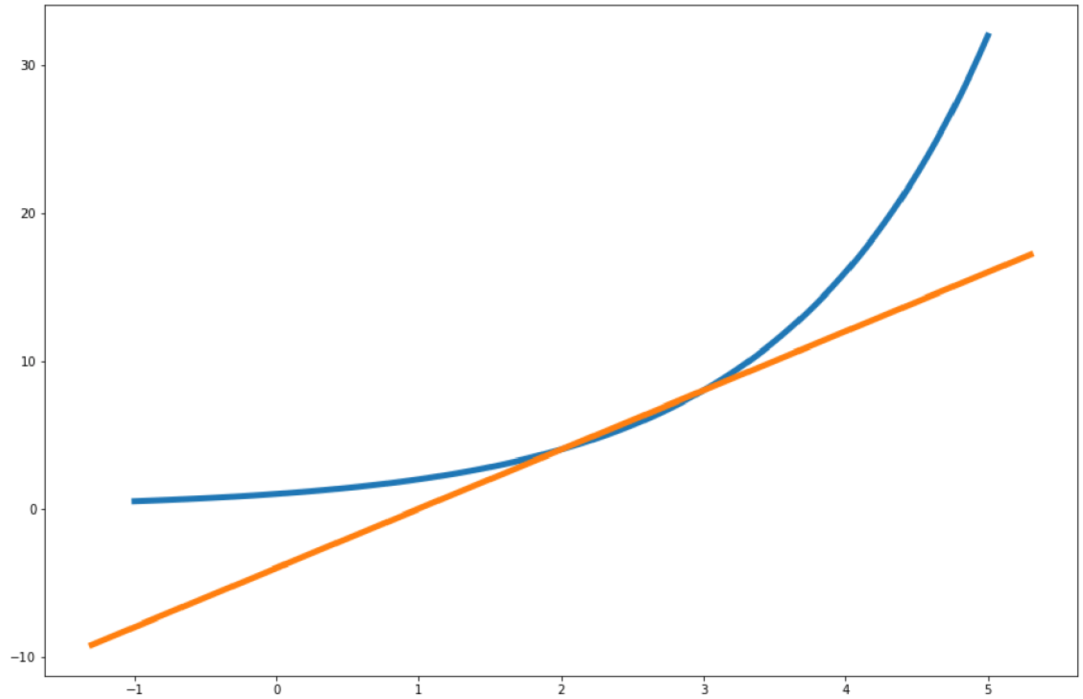
Prikazujemo prvo analitičko rešenje - tangentu u tački $x = 2$

In [16]:
```
fun=lambda x: 2**x


plot_function([0,4],fun)
plot_tangent([2,fun(2)],2**2*np.log(2))
```



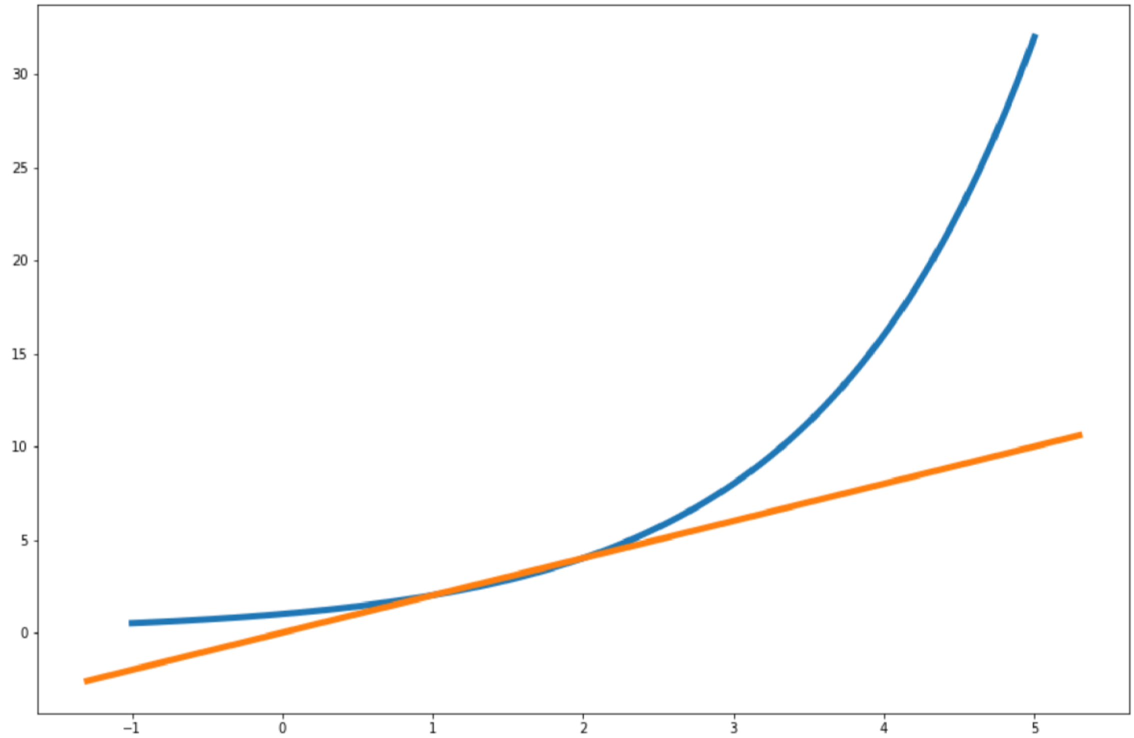
Razlika unapred

In [17]:
```
plot_function([0,4],fun)
h=1 #veliko h da bi se bolje videle razlike izmedju anlitickog i numerickog resenja

plot_tangent([2,fun(2)],razlika_unapred(2,h,fun))
```



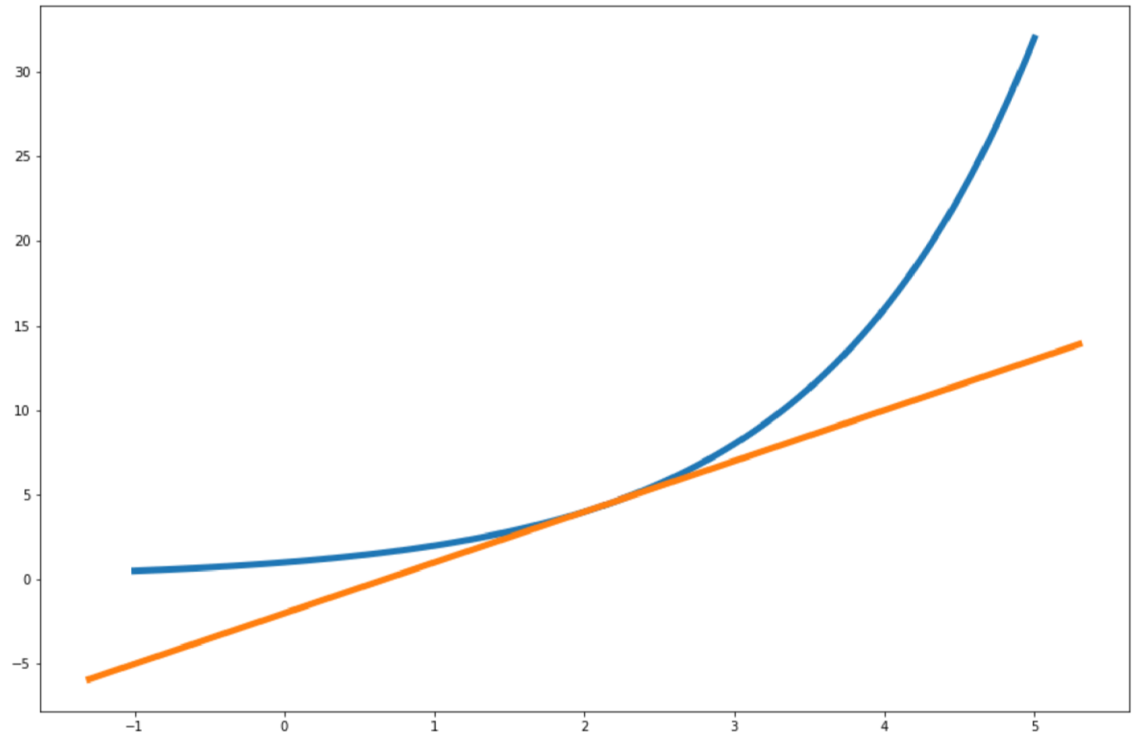
Razlika unazad

In [18]:
```
plot_function([0,4],fun)
plot_tangent([2,fun(2)],razlika_unazad(2,h,fun))
```



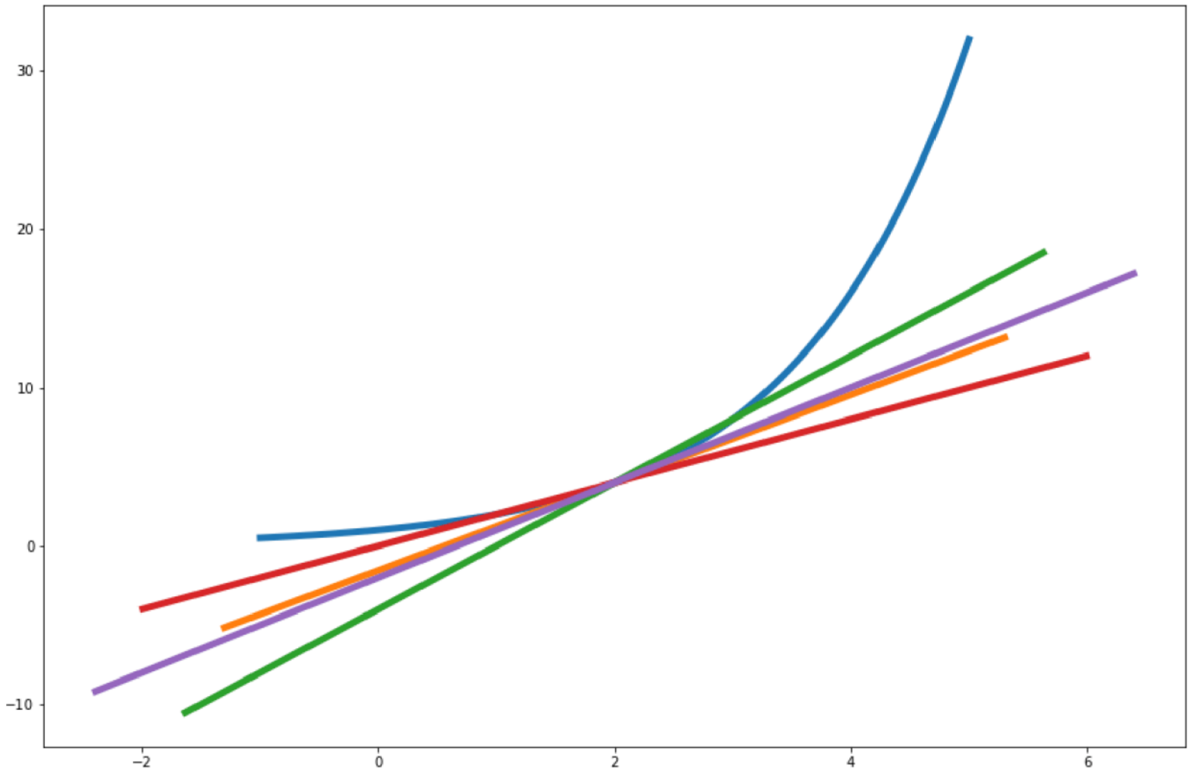
Centralna razlika

In [19]:
```
plot_function([0,4],fun)
plot_tangent([2,fun(2)],centralna_razlika(2,h,fun))
```



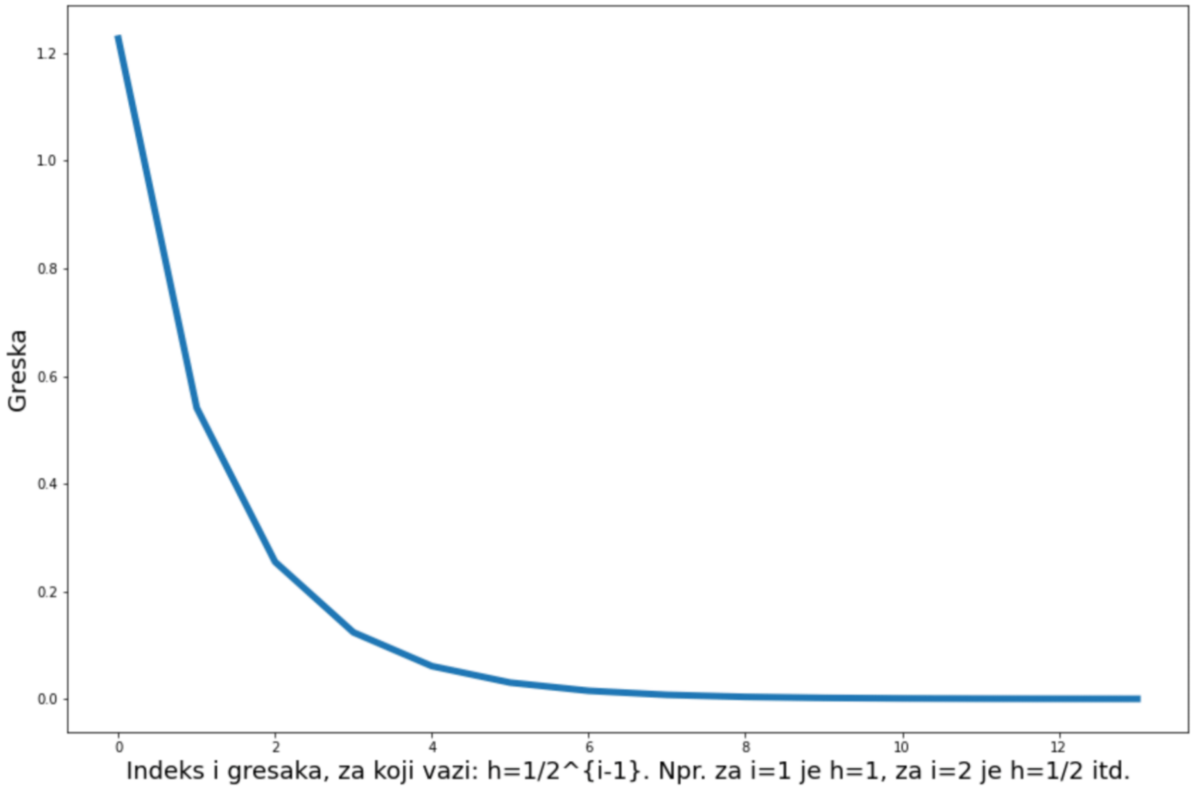
Sve tri konačne razlike i analitičko rešenje

In [20]:
```
plot_function([0,4],fun)

plot_tangent([2,fun(2)],2**2*np.log(2))  #plavo
plot_tangent([2,fun(2)],razlika_unapred(2,h,fun))#zeleno
plot_tangent([2,fun(2)],razlika_unazad(2,h,fun))#crveno
plot_tangent([2,fun(2)],centralna_razlika(2,h,fun))#cyan
```



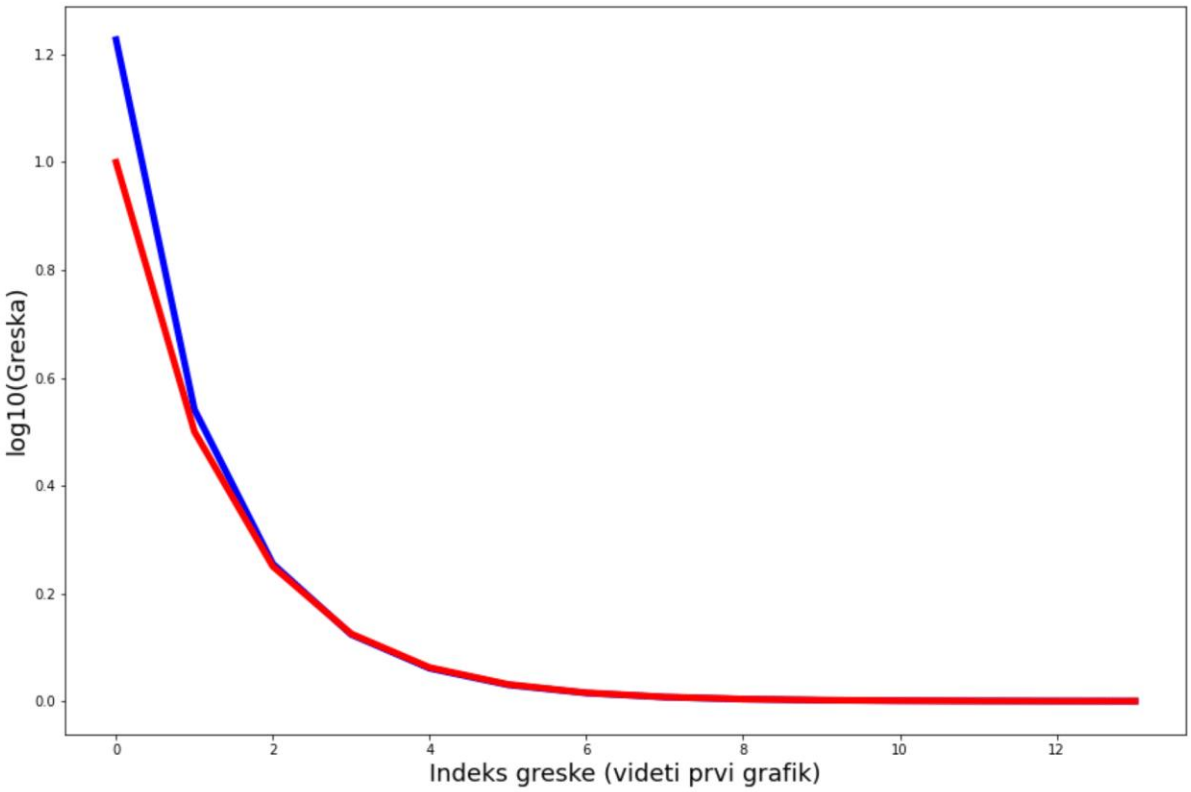
Prikazujemo sada redove grešaka konačnih razlika na primeru koji smo do sada radili. Prvo prikazujemo grešku za razliku unapred, pa za razliku unazad i na kraju za centralnu razliku. Za svaku konačnu razliku prvo crtamo kako greška opada sa smanjenjem h, a nakon toga poredimo funkciju sa njenjim redom greške.

In [21]:
```
fun=lambda x: 2**x
x0=2
[errors_runp,sub_intervals]=calculate_error(x0,fun,0.0001,razlika_unapred,2**2*np.log(2))
```



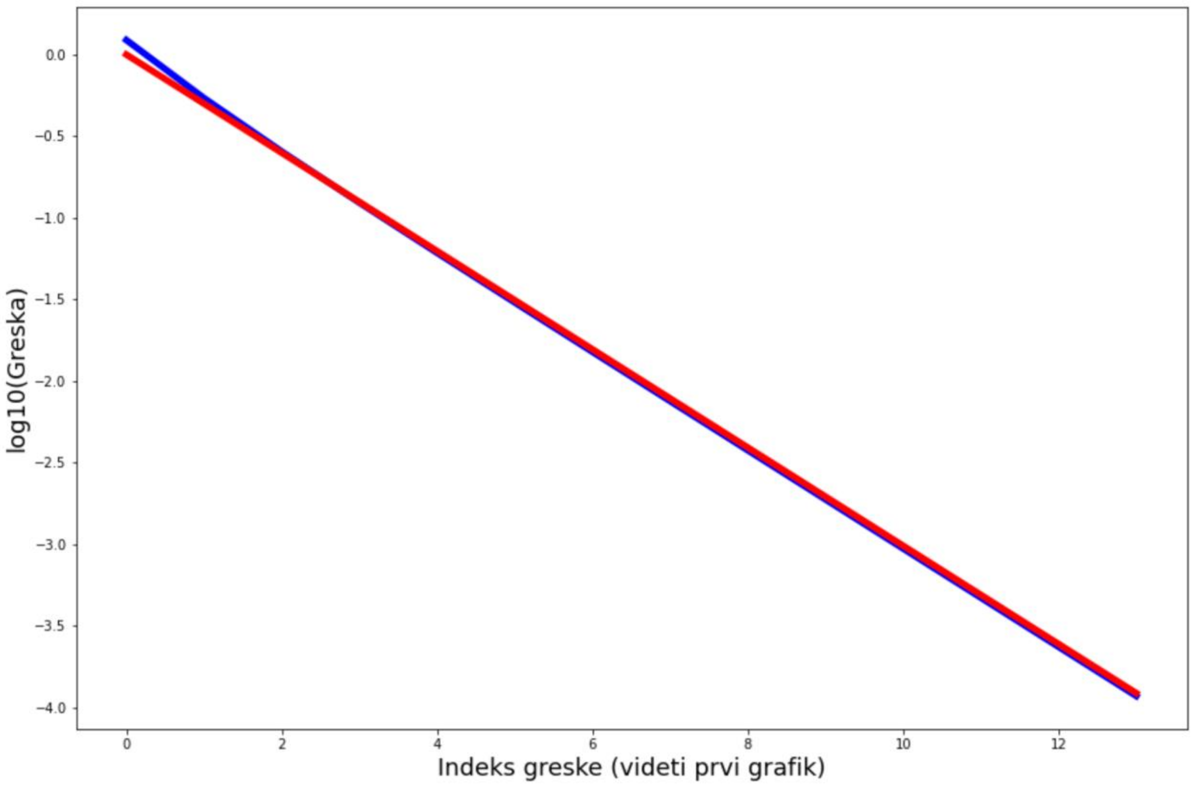
In [22]:
```
plt.plot(np.arange(len(errors_runp)),errors_runp,linewidth=5,color='b')
plt.plot(np.arange(len(sub_intervals)),np.array(sub_intervals),linewidth=5,color='red')
plt.xlabel('Indeks greske (videti prvi grafik)',fontsize=18)
plt.ylabel('log10(Greska)',fontsize=18)
```

Out[22]: `Text(0, 0.5, 'log10(Greska)')`

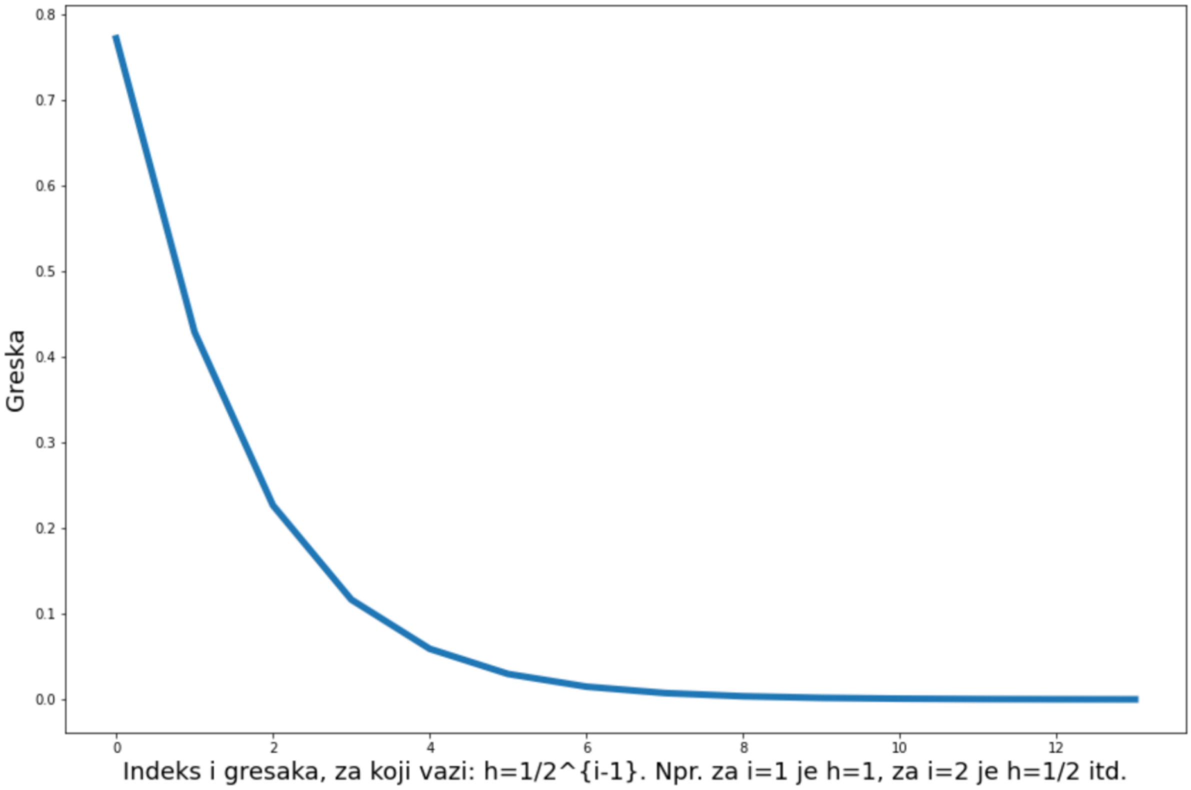


In [23]:
```
plt.plot(np.arange(len(errors_runp)),np.log10(errors_runp),linewidth=5,color='b')
plt.plot(np.arange(len(sub_intervals)),np.log10(np.array(sub_intervals)),linewidth=5,color='red')
plt.xlabel('Indeks greske (videti prvi grafik)',fontsize=18)
plt.ylabel('log10(Greska)',fontsize=18)
```

Out[23]:
```
Text(0, 0.5, 'log10(Greska)')
```

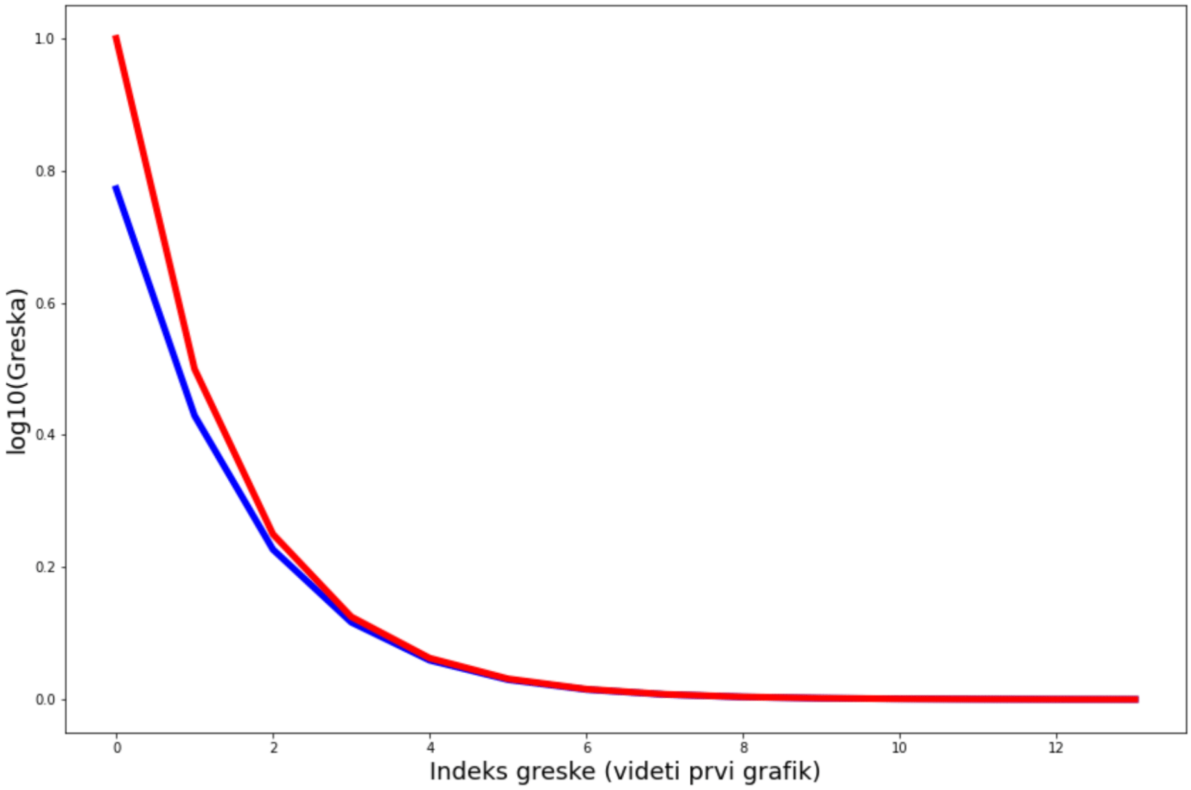


In [24]:
```
[errors_runz,sub_intervals]=calculate_error(x0,fun,0.0001,razlika_unazad,2**2*np.log(2))
```



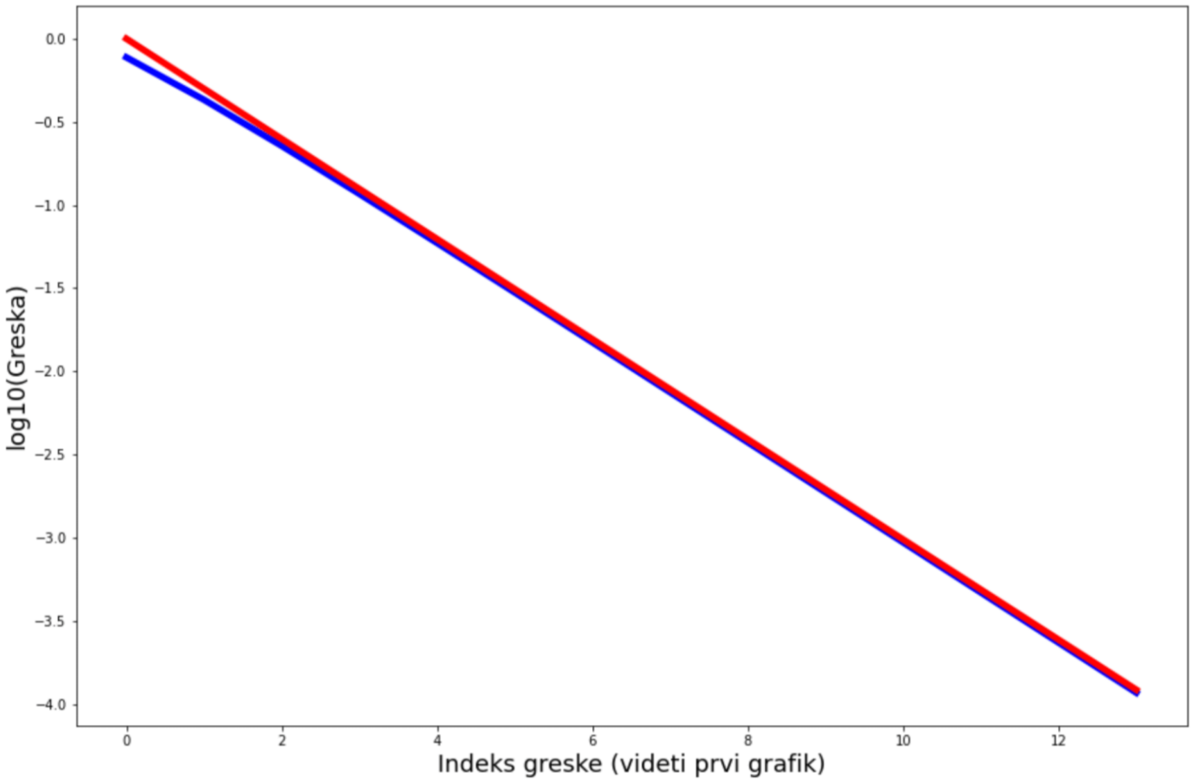
In [25]:
```
plt.plot(np.arange(len(errors_runz)),errors_runz,linewidth=5,color='b')
plt.plot(np.arange(len(sub_intervals)),np.array(sub_intervals),linewidth=5,color='red')
plt.xlabel('Indeks greske (videti prvi grafik)',fontsize=18)
plt.ylabel('log10(Greska)',fontsize=18)
```

Out[25]: `Text(0, 0.5, 'log10(Greska)')`

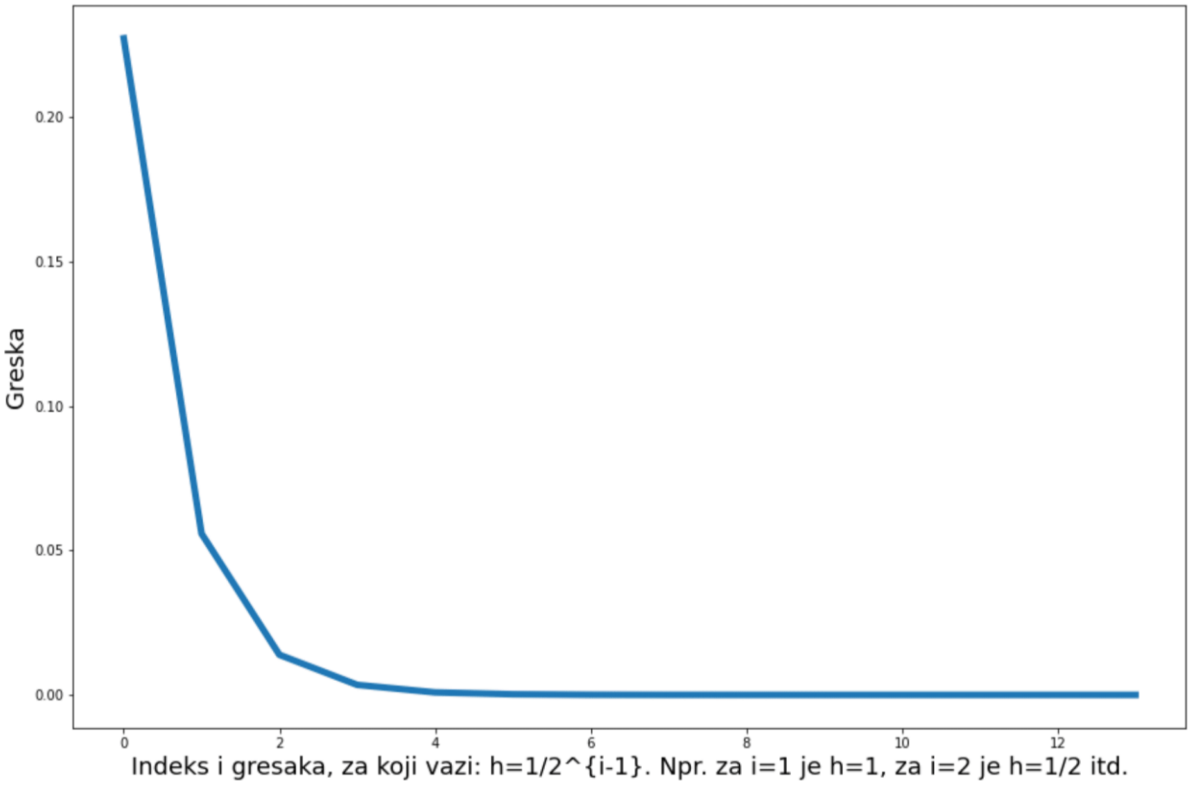


In [26]:
```
plt.plot(np.arange(len(errors_runz)),np.log10(errors_runz),linewidth=5,color='b')
plt.plot(np.arange(len(sub_intervals)),np.log10(np.array(sub_intervals)),linewidth=5,color='red')
plt.xlabel('Indeks greske (videti prvi grafik)',fontsize=18)
plt.ylabel('log10(Greska)',fontsize=18)
```

Out[26]: `Text(0, 0.5, 'log10(Greska)')`

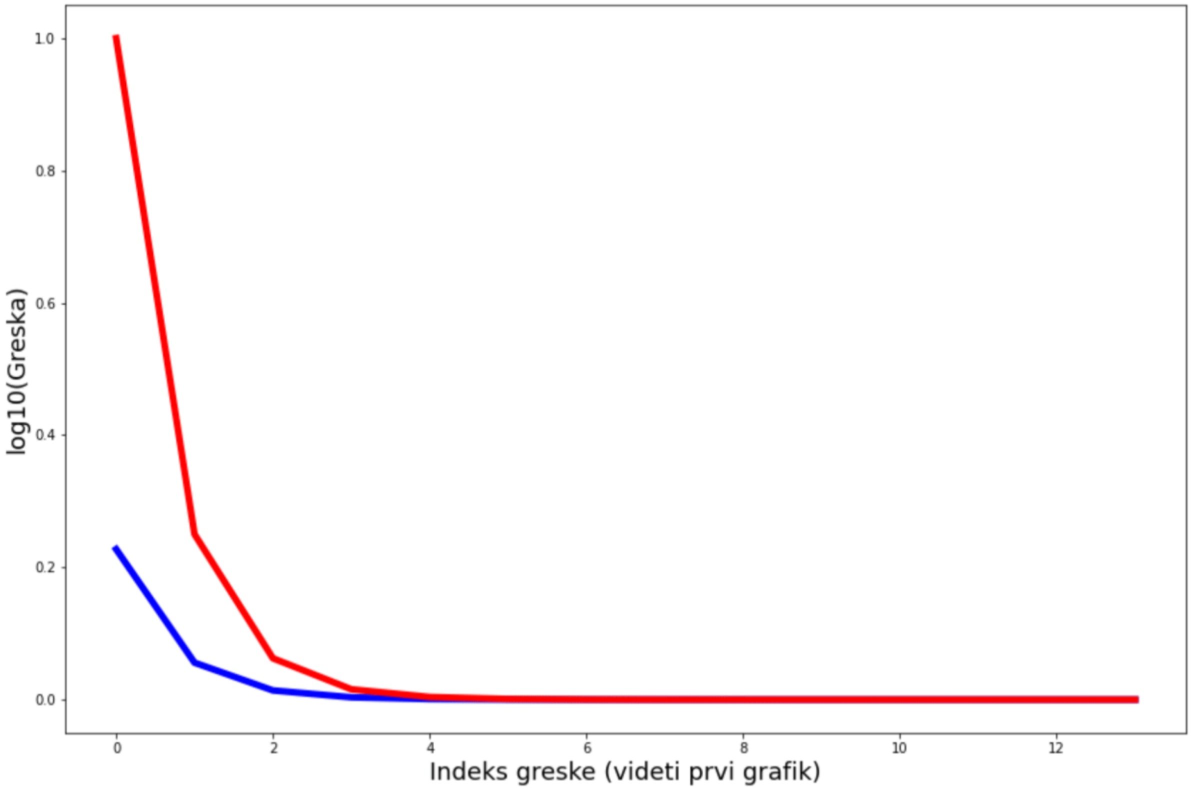


In [27]:
```
[errors_rcen,sub_intervals]=calculate_error(x0,fun,0.0001,centralna_razlika,2**2*np.log(2))
```



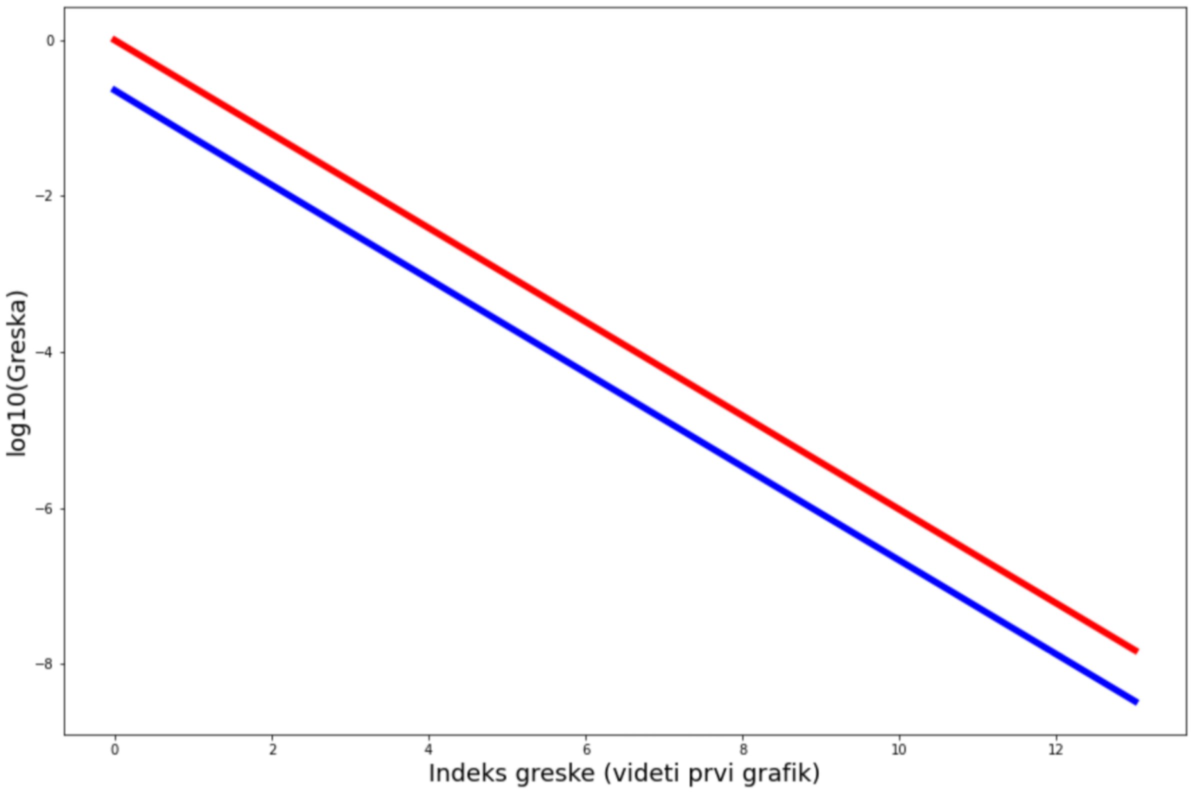
In [28]:
```
plt.plot(np.arange(len(errors_rcen)),errors_rcen,linewidth=5,color='b')
plt.plot(np.arange(len(sub_intervals)),np.array(sub_intervals)**2,linewidth=5,color='red')
plt.xlabel('Indeks greske (videti prvi grafik)',fontsize=18)
plt.ylabel('log10(Greska)',fontsize=18)
```

Out[28]:
```
Text(0, 0.5, 'log10(Greska)')
```



In [29]:
```
plt.plot(np.arange(len(errors_rcen)),np.log10(errors_rcen),linewidth=5,color='b')
plt.plot(np.arange(len(sub_intervals)),np.log10(np.array(sub_intervals)**2),linewidth=5,color='red')
plt.xlabel('Indeks greske (videti prvi grafik)',fontsize=18)
plt.ylabel('log10(Greska)',fontsize=18)
```

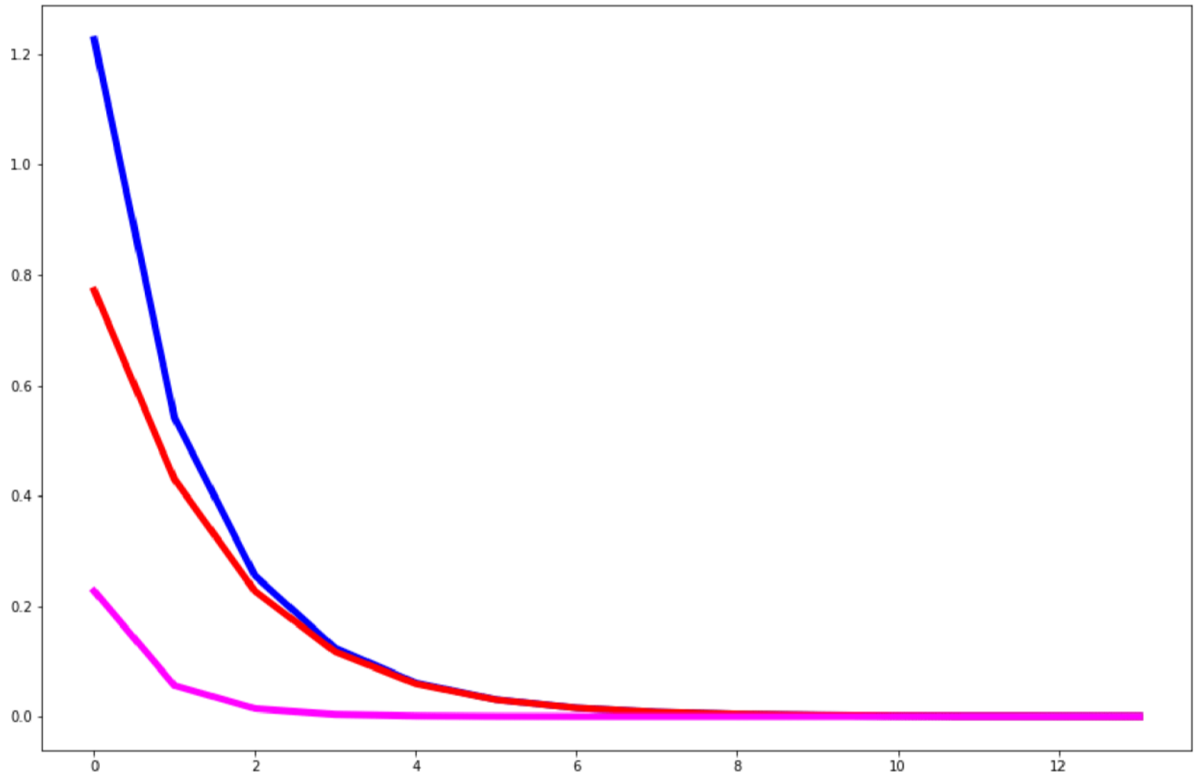
Out[29]: Text(0, 0.5, 'log10(Greska)')



Poredimo sve tri konačne razlike i njihove redove grešaka

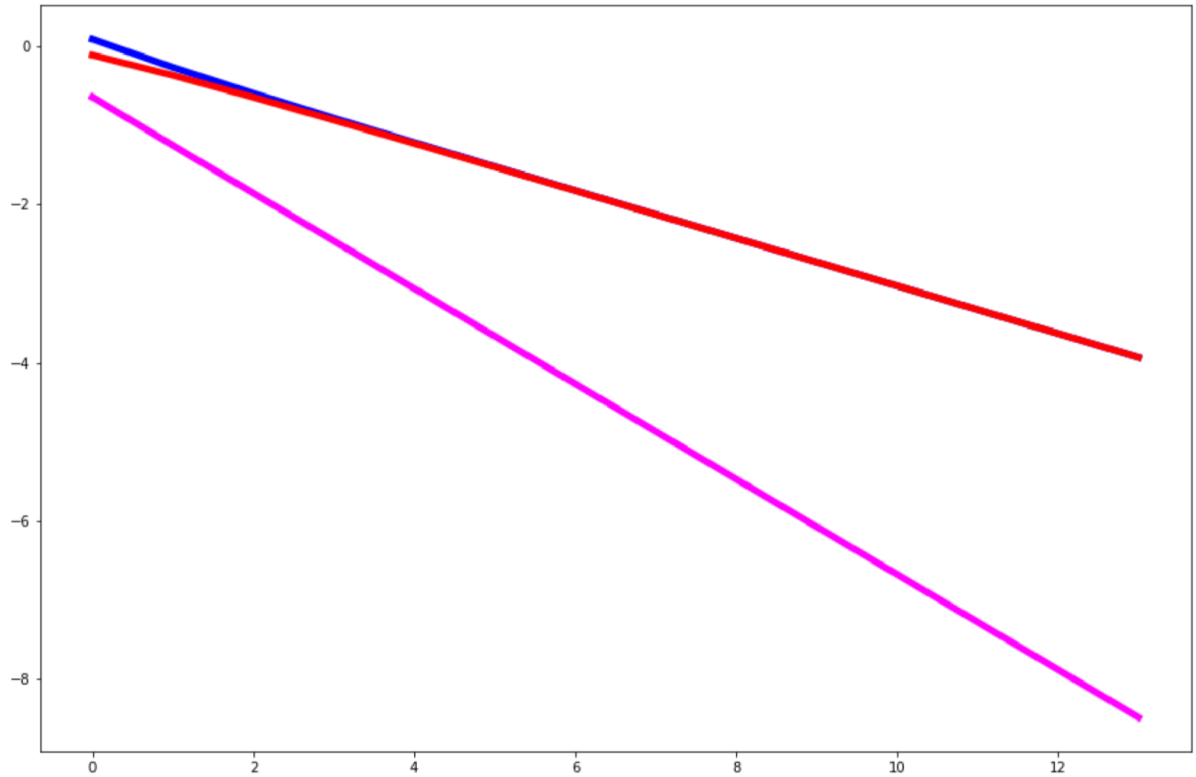
In [30]:
```
plt.plot(np.arange(len(errors_rcen)),errors_runp,linewidth=5,color='b')
plt.plot(np.arange(len(errors_rcen)),errors_runz,linewidth=5,color='red')
plt.plot(np.arange(len(errors_rcen)),errors_rcen,linewidth=5,color='magenta')
```

Out[30]: [<matplotlib.lines.Line2D at 0x189cc24de10>]



In [25]:
```
plt.plot(np.arange(len(errors_rcen)),np.log10(errors_runp),linewidth=5,color='b')
plt.plot(np.arange(len(errors_rcen)),np.log10(errors_runz),linewidth=5,color='red')
plt.plot(np.arange(len(errors_rcen)),np.log10(errors_rcen),linewidth=5,color='magenta')
```

Out[25]: [<matplotlib.lines.Line2D at 0x21072714c88>]



Konačne razlike za izvode višeg reda

Na sličan način kao u slučaju prvog izvoda, izvode se formule i za izvode višeg reda.

Na primer, formula za centralnu razliku za drugi izvod izvodi se na sledeći način:

$$f(x_0+h) = f(x_0) + f'(x_0)h + \frac{1}{2}f''(x_0)h^2 + \frac{1}{6}f^{(3)}(x_0)h^3 + \frac{1}{24}f^{(4)}(x_0)h^4$$

$$f(x_0-h) = f(x_0) - f'(x_0)h + \frac{1}{2}f''(x_0)h^2 - \frac{1}{6}f^{(3)}(x_0)h^3 + \frac{1}{24}f^{(4)}(x_0)h^4$$

Sabiramo prethodna dva Tejlorova razvoja:

$$f(x_0+h) + f(x_0-h) = 2f(x_0) + 2\frac{1}{2}f''(x_0)h^2 + \frac{1}{24}f^{(4)}(x_0)h^4$$

$$f(x_0+h) - 2f(x_0) + f(x_0-h) = f''(x_0)h^2 + \frac{1}{24}f^{(4)}(x_0)h^4$$

$$f''(x_0) = \frac{f(x_0+h) - 2f(x_0) + f(x_0-h)}{h^2} - \frac{1}{24}f^{(4)}(x_0)h^2$$

Što nam daje formulu:

$$f''(x_0) = \frac{f(x_0+h) - 2f(x_0) + f(x_0-h)}{h^2} - O(h^2)$$

Demonstriraćemo upotrebu centralne razlike za drugi izvod na primeru tela u slobodnom padu. Prvi izvod brzine je ubrzanje. Prvi izvod promene položaja tela je brzina. Iz toga sledi da je drugi izvod položaja tela ubrzanje.

Pošto smo u našem primeru postavili otpor vazduha (c) na 0, ubrzanje koje bi trebalo da dobijemo je $9.8\frac{m}{s^2}$.

In [33]:
```
m = 1
c = 0
h0 = 100
v0 = 0
step = 1
t0 = 0
t1 = 4
[time, velocity, height]=simulate(m,c,v0,h0,t0,t1,step)

print(time)
print(velocity)
```

```
[0 1 2 3 4]
[ 0.   9.8 19.6 29.4 39.2]
```

Računamo prvi izvod brzine i tako dobijamo ubrzanje tela.

In [32]:
```
n=len(velocity)
step=1
diff=np.zeros(n-1)

for i in range(n-1):
    v_now=velocity[i]
    v_next=velocity[i+1]
    diff[i]=( v_next - v_now ) / step

print(velocity)
print(diff)
```

```
[ 0.   9.8 19.6 29.4 39.2]
[9.8 9.8 9.8 9.8]
```

Računamo drugi izvod promene položaja tela i tako dobijamo ubrzanje tela.

In [34]:
```
height=-height
diff=np.zeros(n-2)

for i in range(1,n-1):
    h_before=height[i-1]
    h_now=height[i]
    h_next=height[i+1]
    diff[i-1]=( h_next-2*h_now+h_before ) / (step**2)

print(diff)
print(height)
```

```
[9.8 9.8 9.8]
[-100.   -90.2  -70.6  -41.2   -2. ]
```

Primena Ričardsonove ekstrapolacije na numeričko diferenciranje

Kao i kod Rombergove integracije, primenom Ričardsonove ekstrapolacije moguće je kombinovati dve formule manjeg reda tačnosti tako da dobijemo rezultat većeg reda tačnosti.

Postupak primene Ričardsonove ekstrapolacije demonstriraćemo pomoću centralne razlike za prvi izvod.

Posmatramo centralnu razliku kao funkciju od koraka h, pri čemu su x i $f(x)$ fiksirani:

$$\varphi(h) = \frac{f(x+h) - f(x-h)}{2h} - O(h^2)$$

Podsetićemo se na koji način smo izveli centralnu razliku da bi se faktori grešaka jasnije videli:

$$f(x_0+h) = f(x_0) + f'(x_0)h + \frac{1}{2}f''(x_0)h^2 + \frac{1}{6}f^{(3)}(x_0)h^3 + \frac{1}{4!}f^{(4)}(x_0)h^4 + \frac{1}{5!}f^{(5)}(x_0)h^5$$

$$f(x_0-h) = f(x_0) - f'(x_0)h + \frac{1}{2}f''(x_0)h^2 - \frac{1}{6}f^{(3)}(x_0)h^3 + \frac{1}{4!}f^{(4)}(x_0)h^4 - \frac{1}{5!}f^{(5)}(x_0)h^5$$

Oduzimamo predhodna dva Tejlorova razvoja:

$$f(x_0+h) - f(x_0-h) = f(x_0) + 2f'(x_0)h + 2\frac{1}{6}f^{(3)}(x_0)h^3 + 2\frac{1}{5!}f^{(5)}(x_0)h^5$$

Što nam daje formulu:

$$f'(x_0) = \frac{f(x_0+h) - f(x_0-h)}{2h} - \frac{1}{3}f^{(3)}(x_0)h^2 - \frac{2}{5!}f^{(5)}(x_0)h^4$$

Iz predhodnog možemo da zaključimo da je greška primene centralne razlike oblika:

$$f'(x_0) = \frac{f(x_0+h) - f(x_0-h)}{2h} - ah^2 - bh^4 - ch^6 - dh^8 - \dots$$

gde su $a, b, c, d \dots$ konstante.

Takođe možemo da zaključimo da je najveći faktor greške reda h^2, prvi sledeći h^4 itd.

To znači da ako pomoću Ričardsonove ekstrapolacije uklonimo najveći faktor greške centralne razlike, dobićemo rezultat formule reda $O(h^4)$

Primenjujemo sada Ričardsonovu ekstrapolaciju. Ako su x i $f(x)$ fiksirani pokazali smo da se centralna razlika od tačnog rešenja (prvog izvoda dobijenog analitički) razlikuje za grešku:

$$\varphi(h) = f'(x_0) + ah^2 + bh^4 + ch^6 + dh^8 + \dots$$

Koristimo sada $\varphi(h)$ i $\varphi(\frac{h}{2})$

$$\varphi(h) = f'(x_0) + ah^2 + bh^4 + \ldots$$

$$\varphi\left(\frac{h}{2}\right) = f'(x_0) + a\left(\frac{h}{2}\right)^2 + b\left(\frac{h}{2}\right)^4 + \ldots$$

$$\varphi\left(\frac{h}{2}\right) = f'(x_0) + a\left(\frac{h^2}{4}\right) + b\left(\frac{h^4}{16}\right) + \ldots$$

Izračunaćemo sada: $\varphi(h) - 4\varphi\left(\frac{h}{2}\right)$

$$\varphi(h) - 4\varphi\left(\frac{h}{2}\right) = f'(x_0) - 4f'(x_0) + ah^2 - 4a\left(\frac{h^2}{4}\right) + bh^4 - 4b\left(\frac{h^4}{16}\right)$$

$$\varphi(h) - 4\varphi\left(\frac{h}{2}\right) = -3f'(x_0) + b\left(\frac{3}{4}\right)h^4$$

$$3f'(x_0) = 4\varphi\left(\frac{h}{2}\right) - \varphi(h) + b\left(\frac{3}{4}\right)h^4$$

$$f'(x_0) = \frac{4\varphi\left(\frac{h}{2}\right) - \varphi(h)}{3} + O(h^4)$$

Dobili smo način da iskombinujemo $\varphi(h)$ i $\varphi(\frac{h}{2})$ tako da dobijemo tačnost reda $O(h^4)$.

Na sličan način kao kod robergove integracije možemo da izvedemo formulu za kombinovanje konačnih razlika većeg reda tačnosti:

$$\varphi(h)_{i,j} = \frac{4^j\varphi(h)_{i,j+1} - \varphi(h)_{i,j}}{4^{j-1}}$$

gde je i red tačnosti metoda, a povećanje j je smanjenje koraka h dva puta.

Primenićemo sada Ričardsonovu ekstrapolaciju na primer od ranije:

$$f'(2^x) = ?$$
$$x = 2$$

Analitičko rešenje je:

$$f'(2^x) = 2^x \cdot ln(2)$$

$$f'(2) = 2^2 \cdot ln(2) = 2.7726$$

Centralna razlika za $h = 0.5$:

$$f'(x_0) = \frac{f(x_0+h) - f(x_0-h)}{2h} = \frac{2^{2+0.5} - 2^{2-0.5}}{2 \cdot 0.5} = 2.8284$$

$$E_R = \left|\frac{2.7726 - 2.8284}{2.7726}\right| = 0.0201 = 2.01\%$$

Koristimo sada $h = 0.25$.

Centralna razlika

$$f'(x_0) = \frac{f(x_0+h) - f(x_0-h)}{2h} = \frac{2^{2+0.25} - 2^{2-0.25}}{2 \cdot 0.25} = 2.7865$$

$$E_R = \left| \frac{2.7726 - 2.7865}{2.7726} \right| = 0.005 = 0.5\%$$

Primena Ričardsonove ekstrapolacije

$$\frac{4\varphi\left(\frac{h}{2}\right) - \varphi(h)}{3} = \frac{4 \cdot 2.7865 - 2.8284}{3} = 2.7725$$

$$E_R = \left| \frac{2.7726 - 2.7725}{2.7726} \right| = 0.000036 = 0.0036\%$$

Rešenje koje smo dobili ima relativnu grešku od približno 0.0036%, što je značajno manje nego 0.5%, koliko smo imali za centralnu razliku sa $h = 0.25$. U udžbeniku je pokazano da dobijeni rezultat 2.7725 odgovara rezultatu koji bi se dobio primenom formule za konačnu razliku koja je reda greške $O(h^4)$.

U nastavku je dat kod primene Ričardsonove ekstrapolacije na centralnu razliku. Kod je vrlo sličan kodu za Rombergovu integraciju.

In [70]:
```
def diff_ricardson(x0,n,funkcija):
    A=np.zeros([n,n])
    for i in range(n):
        A[i,0]=centralna_razlika(x0,(1/(2**i)),funkcija)

    for j in range(1,n):
        for i in range(n-j):
            A[i,j]=( 4**j * A[i+1,j-1] - A[i,j-1] ) / ( (4**j) - 1 )

    print(A)
    return A[0,n-1]
```

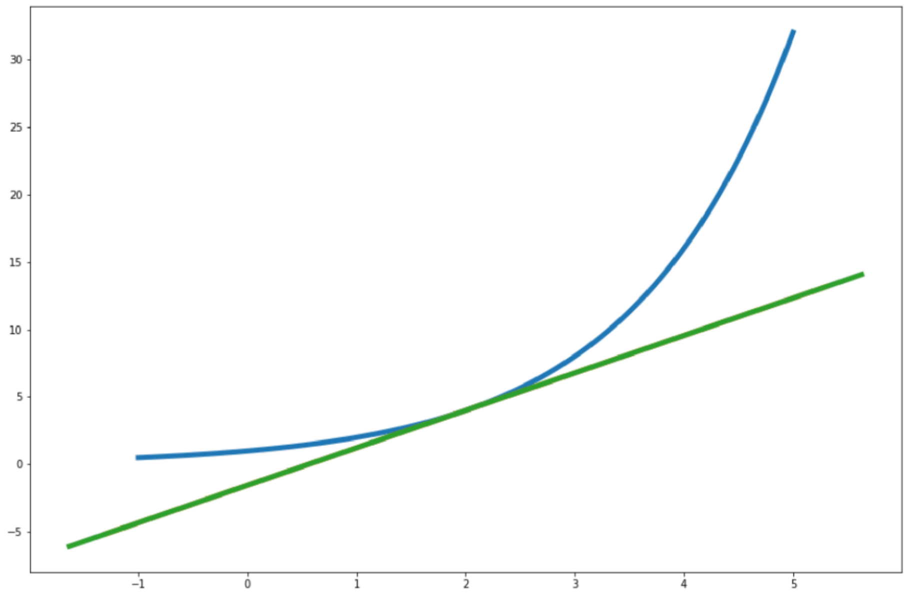
In [71]:
```
fun=lambda x:2**x
x0=2
n=5
d=diff_ricardson(x0,n,fun)
```

```
[[3.         2.77123617 2.77258968 2.77258872 2.77258872]
 [2.82842712 2.77250509 2.77258874 2.77258872 0.        ]
 [2.7864856  2.77258351 2.77258872 0.         0.        ]
 [2.77605903 2.7725884  0.         0.         0.        ]
 [2.77345606 0.         0.         0.         0.        ]]
```

Grafički poredimo tačno rešenje i rešenje dobijeno pomoću fukcije *diff_ricardson*.

In [72]:
```
h=1 #veliko h da bi se bolje videle razlike izmedju anlitickog i numerickog resenja

plot_function([0,4],fun)

plot_tangent([2,fun(2)],diff_ricardson(x0,n,fun))

plot_tangent([2,fun(2)],2**2*np.log(2))
```

```
[[3.         2.77123617 2.77258968 2.77258872 2.77258872]
 [2.82842712 2.77250509 2.77258874 2.77258872 0.        ]
 [2.7864856  2.77258351 2.77258872 0.         0.        ]
 [2.77605903 2.7725884  0.         0.         0.        ]
 [2.77345606 0.         0.         0.         0.        ]]
```



Vidimo da je razlika između tačnog rešenja i rešenja dobijenog numeričkom metodom toliko mala da je perimenta na grafiku.