VI.9.32

огласно переписи население США менялось следующим обра-зом:

1910 – 92 228 496 человек,

1920 – 106 021 537,

1930 – 123 202 624,

1940 – 132 164 569,

1950 – 151 325 798,

1960 – 179 323 175,

1970 – 203 211 926,

1980 – 226 545 805,

1990 – 248 709 873,

2000 – 281 421 906.

In [1]:

```
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt
```

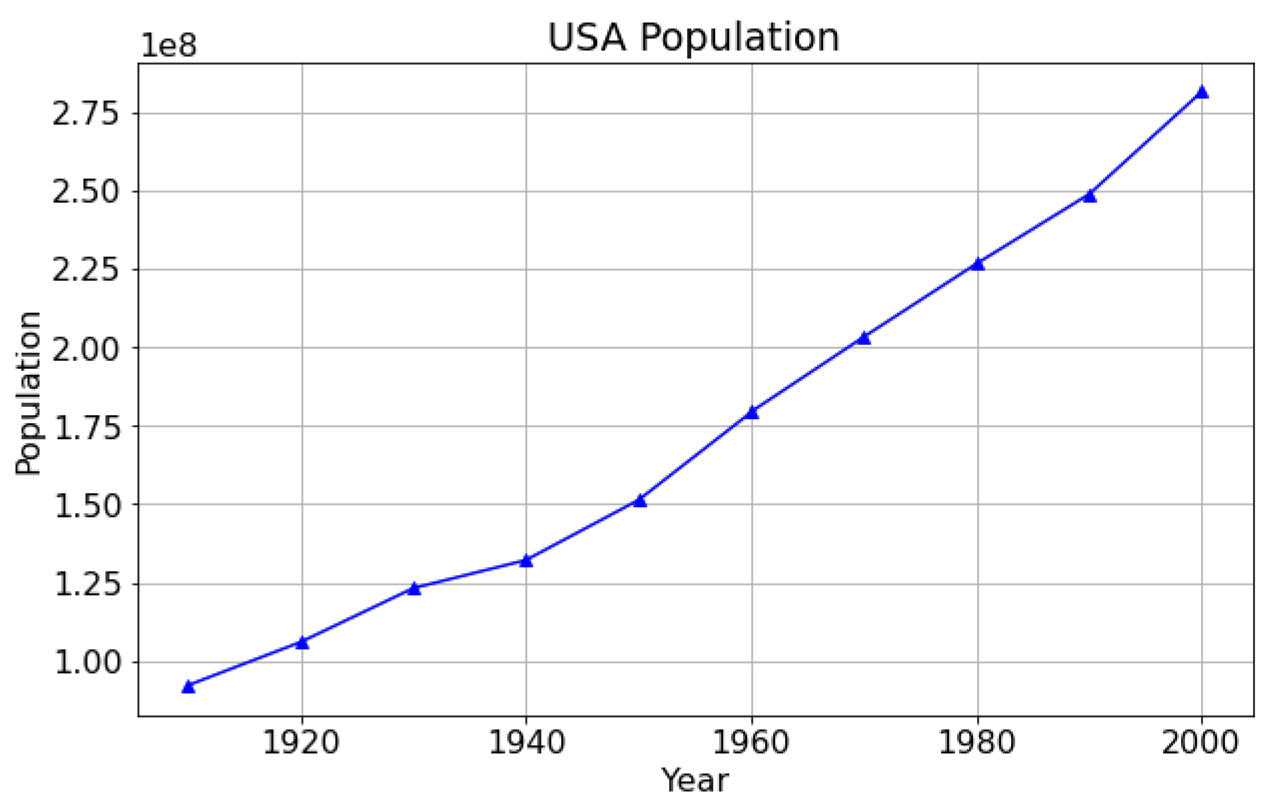
а) По приведенным данным построить интерполянт в форме Нью-тона. Вычислить экстраполированное значение численности населения США в 2010 году и сравнить с точным значением 308 745 538 человек.

Строим таблицу разделённый разностей

In [2]:

```
x  = np.array([1910, 1920, 1930, 1940, 1950, 1960, 1970, 1980, 1990, 2000])
b0 = np.array([92228496, 106021537, 123202624, 132164569, 151325798, 179323175, 20321

mpl.rcParams['font.size']=16
plt.figure(figsize=(10,6))
plt.plot(x, b0,"b-^")
plt.title("USA Population")
plt.ylabel("Population")
plt.xlabel("Year")
plt.grid(b=True, which='major', axis='both')
plt.grid(b=True, which='minor', axis='both')
plt.show()
```



In [3]:

```
bnprev = b0.copy()
bnnext = b0.copy()

lastCoeffInterpoll = np.array([b0[-1]])
```

In [4]:

```
step = 1

while(bnnext.size > 1):
    bnnext = np.array([])
    for i in range(bnprev.size - 1):
        elem = (bnprev[i+1] - bnprev[i]) / (x[i + step] - x[i])
        bnnext = np.append(bnnext, elem)
    bnprev = bnnext.copy()
    step = step + 1

    lastCoeffInterpoll = np.append(lastCoeffInterpoll, bnnext[-1])
lastCoeffInterpoll
```

Out[4]:

```
array([2.81421906e+08, 3.27120330e+06, 5.27398250e+04, 1.95296267e+03,
       5.13863125e+01, 1.37511733e+00, 5.16230958e-02, 1.70468286e-03,
       3.83292770e-05, 5.09932804e-07])
```

In [5]:

```
def getNewtonInterpol(coeffs, x, x0):
    ans = coeffs[0]
    for i in range(1, coeffs.size):
        prodx = 1
        for j in range(i):
            prodx *= (x0 - x[x.size - j - 1])
        ans += prodx * coeffs[i]
    return ans
```

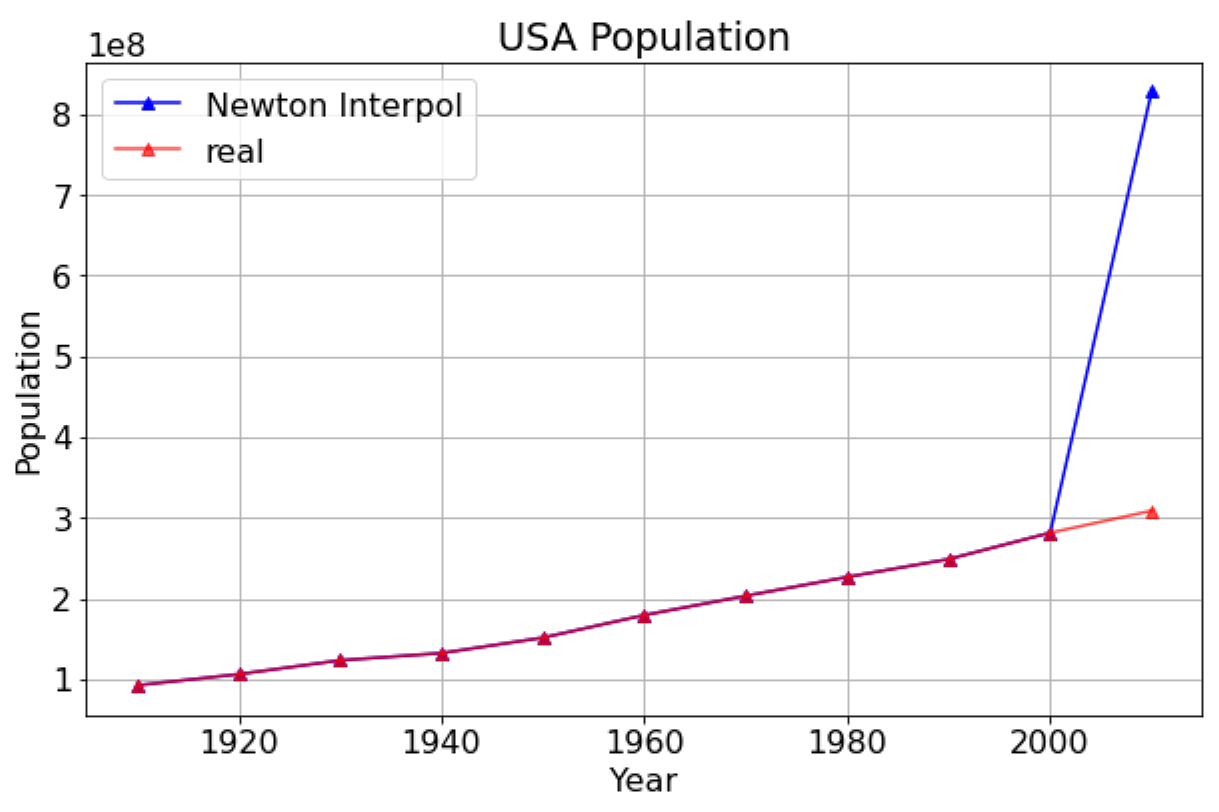
In [6]:

```
interpollData = np.array([])
years = np.linspace(1910, 2000, 10)
approxYears = np.linspace(1910, 2010, 11)
for i in range(approxYears.size):
    interpollData = np.append(interpollData, getNewtonInterpol(lastCoeffInterpoll, yea

mpl.rcParams['font.size']=16
plt.figure(figsize=(10,6))
plt.plot(approxYears, interpollData,"b-^", label="Newton Interpol")

x  = np.array([1910, 1920, 1930, 1940, 1950, 1960, 1970, 1980, 1990, 2000, 2010])
b0 = np.array([92228496, 106021537, 123202624, 132164569, 151325798, 179323175, 20321
plt.plot(x, b0,"r-^", label="real", alpha = 0.7)

plt.title("USA Population")
plt.ylabel("Population")
plt.xlabel("Year")
plt.grid(b=True, which='major', axis='both')
plt.grid(b=True, which='minor', axis='both')
plt.legend()
plt.show()
```



In [7]:

```
print(getNewtonInterpol(lastCoeffInterpoll, years, 2010))
```

```
827906509.0000001
```

б) По этим же данным построить сплайн-аппроксимацию, экстрапо-лировать данные на 2010 год, сравнить с точным значением. Какие до-полнительные условия для построения сплайна нужно поставить в этом случае?

Построим систему для нахождения с коэффициентов

In [21]:

```
x  = np.array([1910, 1920, 1930, 1940, 1950, 1960, 1970, 1980, 1990, 2000])
y = np.array([92228496, 106021537, 123202624, 132164569, 151325798, 179323175, 203211

b = np.zeros([x.size])
for i in range(1, x.size - 1):
    b[i] = (y[i + 1] - y[i]) / 10.0 - (y[i] - y[i - 1]) / 10.0
b[0] = 0
b[-1] = 0

A = np.zeros([x.size, x.size])
for i in range(1, x.size - 1):
  for j in range(0, x.size - 1):
    # главная диагональ = 2(H(i+1) + H(i))
    if i == j:
      A[i][j] = 2 * (x[i + 1] - x[i - 1])
    # другие диагонали
    elif i == j + 1:
      A[i][j] = x[i] - x[i - 1]
    elif j == i + 1:
      A[i][j] = x[i + 1] - x[i]

A[0][0]   = 1
A[-1][-1] = 1
```

In [22]:

```
A
```

Out[22]:

```
array([[ 1.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.],
       [10., 40., 10.,  0.,  0.,  0.,  0.,  0.,  0.,  0.],
       [ 0., 10., 40., 10.,  0.,  0.,  0.,  0.,  0.,  0.],
       [ 0.,  0., 10., 40., 10.,  0.,  0.,  0.,  0.,  0.],
       [ 0.,  0.,  0., 10., 40., 10.,  0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0., 10., 40., 10.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.,  0., 10., 40., 10.,  0.,  0.],
       [ 0.,  0.,  0.,  0.,  0.,  0., 10., 40., 10.,  0.],
       [ 0.,  0.,  0.,  0.,  0.,  0.,  0., 10., 40.,  0.],
       [ 0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  1.]])
```

In [23]:

```
b
```

Out[23]:

```
array([      0. ,  338804.6, -821914.2, 1019928.4,  883614.8, -410862.6,
        -55487.2, -116981.1, 1054796.5,       0. ])
```

решим полученную систему методом прогонки

In [24]:

```
def solve(A, f):
    newB = np.zeros(f.size - 1)
    newD = np.zeros(f.size - 1)
    for i in range(0, f.size-1):
        w = A[i+1][i] / A[i][i]
        newB[i] = A[i][i] - w * A[i][i+1]
        newD[i] = f[i+1] - w * f[i]

    x = np.zeros(f.size)
    x[-1] = newD[-1] / newB[-1]
    for i in range(f.size-2,-1,-1):
        x[i] = (newD[i] - A[i][i+1] * x[i+1]) / A[i][i]
    return x
```

In [25]:

```
solution = solve(A, b)
solution
```

Out[25]:

```
array([338804.6        , -29081.40894669,  25664.10078674,  19884.29185303,
       -16673.89741211,   3518.95964844,  -9352.99359375,  27101.044375  ,
            0.         ,      0.         ])
```

решим готовыми библиотеками питона:

In [26]:

```
c = np.linalg.solve(A, b)
c
```

Out[26]:

```
array([-9.31322575e-11,  1.64275135e+04, -3.18295940e+04,  2.86994426e+04,
        1.90246637e+04, -1.64366173e+04,  5.63554553e+03, -1.16542848e+04,
        2.92834837e+04,  0.00000000e+00])
```

Не сошлось :(. Значит метод прогонки реализован мной неправильно -> воспользуемся готовым решением.

найдём остальные коэффициенты

In [27]:

```
a = y
k = np.arange(a.size - 1)
d = (c[k+1] - c[k]) / 10.
b = (a[k+1] - a[k]) / 10. + c[k+1] * 10 / 3. + c[k] * 10 / 6.
```

In [28]:

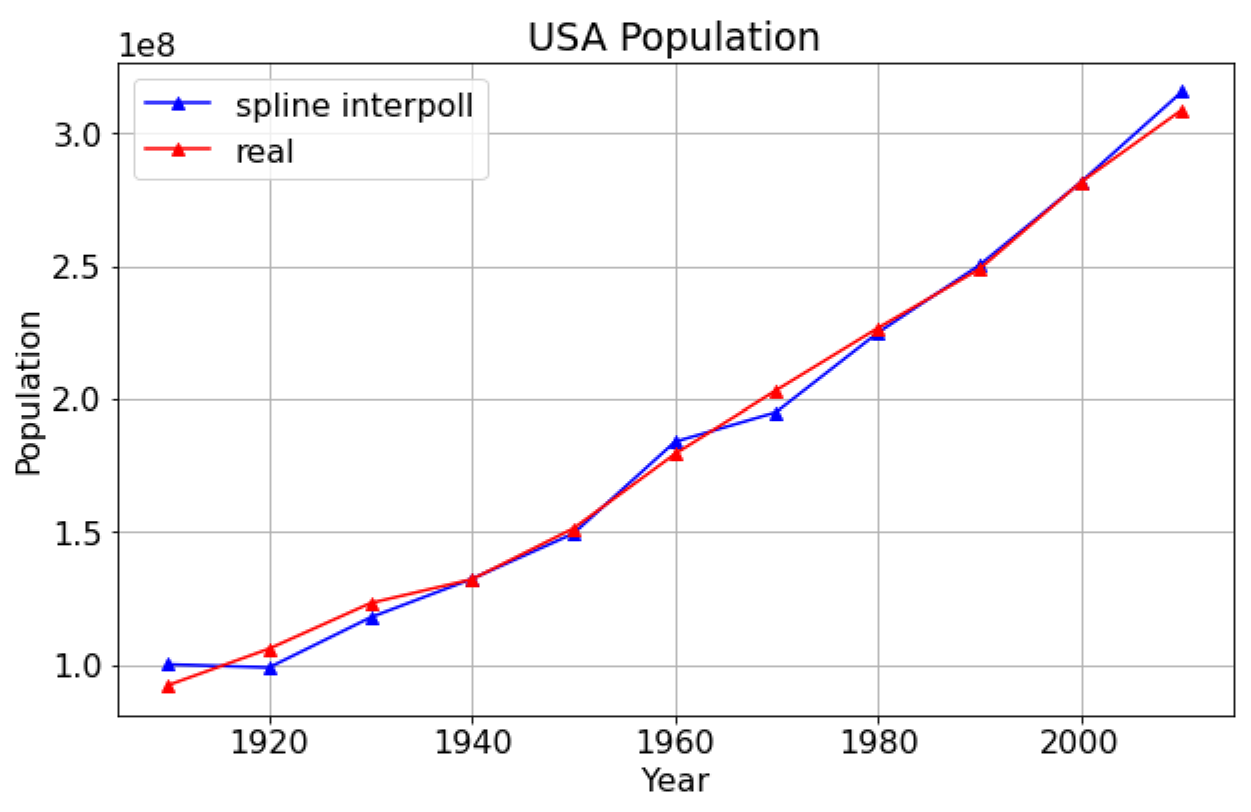
```
import bisect
xApprox = np.linspace(1910, 2010, 11)
yApprox = np.zeros(xApprox.size)
for i in range(xApprox.size):
    # find poly
    x0 = xApprox[i]
    j = bisect.bisect_right(x, x0)
    if (j >= x.size - 1):
        j = x.size - 2
    # interpoll with j poly
    yi = a[j + 1] + b[j] * (x0 - x[j + 1]) + c[j] / 2.0 * (x0 - x[j + 1])**2 + d[j] /
    yApprox[i] = yi

mpl.rcParams['font.size']=16
plt.figure(figsize=(10,6))
plt.plot(xApprox, yApprox,"b-^", label="spline interpoll")
print(yApprox[-1])

x  = np.array([1910, 1920, 1930, 1940, 1950, 1960, 1970, 1980, 1990, 2000, 2010])
b0 = np.array([92228496, 106021537, 123202624, 132164569, 151325798, 179323175, 20321
plt.plot(x, b0,"r-^", label="real")

plt.title("USA Population")
plt.ylabel("Population")
plt.xlabel("Year")
plt.grid(b=True, which='major', axis='both')
plt.grid(b=True, which='minor', axis='both')
plt.legend()
plt.show()
```

```
315598113.18509066
```



Реальное значение: 308 745 538

In [ ]: