Skilaverkefni10. Stærð- og Reiknifræði REI201G

Donn Eunice Bartido deb5@hi.is

mars 2023

38.C Kennitölur og vartöluprófun

- Skv. Wikipediu er kennitala númer á forminu DDMMÁÁ-NNPÖ þar sem DDMMÁÁ er fæðingardagur, -mánuður og -ár, NN er slembitala, P er prófsumma eða vartala, og Ö táknar fæðingaröld. Til að finna vartöluna eru fyrstu átta tölurnar margfaldaðar með tölunum 3, 2, 7, 6, 5, 4, 3 og 2 frá vinstri til hægri, margfeldin eru svo lögð saman og afgangur þegar deilt er í summuna með 11 fundinn. Ef afgangurinn er 0 er vartalan 0, ef hann er 1 er talan ónothæf sem kennitala, en annars fæst vartalan með því að draga afganginn frá 11.

38C.1 [Skrifið fall]

Skrifið fall sem ákvarðar hvort kennitala sem gefin er með 10 stafa streng (án bandstriks) sé með rétta vartölu og skilar svarinu sem rökgildi (True eða False). Prófið með "9999999999" og líka með eigin kennitölu.

In [10]:
```
def er_gild_kennitala(kennitala):
    # athuga að kennitala sé á réttu formi
    if not kennitala.isdigit() or len(kennitala) != 10:
        return False

    # reikna vartölu
    vartala_sum = 0
    vartala_fjoldi = 0
    for i in range(8):
        vartala_sum += int(kennitala[i]) * [3, 2, 7, 6, 5, 4, 3, 2][i]
    vartala_sum += int(kennitala[8]) * 6
    vartala_sum += int(kennitala[9]) * 0

    # finna vartöluna
    afgangur = vartala_sum % 11
    if afgangur == 0:
        vartala = 0
    elif afgangur == 1:
        return False
    else:
        vartala = 11 - afgangur

    # skila rökgildi hvort kennitala sé gild eða ekki
    return int(kennitala[-1]) == vartala
```

In [ ]:
```
  vartala_sum = 0
    vartala_fjoldi = 0
    for i in range(8):
        vartala_sum += int(kennitala[i]) * [3, 2, 7, 6, 5, 4, 3, 2][i]
```

```
    vartala_sum += int(kennitala[8]) * 6
    vartala_sum += int(kennitala[9]) * 0

    # finna vartöluna
    afgangur = vartala_sum % 11
    if afgangur == 0:
        vartala = 0
    elif afgangur == 1:
        return False
    else:
        vartala = 11 - afgangur
```

In [26]:
```
print(er_gild_kennitala("9999999999"))
```

```
False
```

In [19]:
```
print(er_gild_kennitala("2507943479"))
```

```
True
```

38C.2 [Lesa gögn og birta töflu]

Í skránni https://cs.hi.is/python/kennitolur.txt eru skáldaðar upplýsingar um 15 próftaka, prófnúmer, kennitölur og nöfn. Lesið skrána inn í þrjá lista með því að nota np.loadtxt(...) og birtið töflu með innihaldinu.

In [9]:
```
import numpy as np
import requests
import io

# ná í gögn
url = "https://cs.hi.is/python/kennitolur.txt"

# svo ég fái rétt gögn ekki eh garble txt
response = requests.get(url)
text = response.content.decode('utf-8')

# Hlaða textan með np.loadtxt
data = np.loadtxt(io.StringIO(text), delimiter=',', skiprows=1, dtype=str)

print(data)  # do something with the data
```

```
[['Sigrún Jónsdóttir' '0176' '2903993279']
 ['Kristín Fjóludóttir' '0542' '1206972699']
 ['Birta Lárusdóttir' '0970' '2605973109']
 ['Erla Ýr Guðnadóttir' '1419' '1210012330']
 ['Anton Ingi Þórsson' '4854' '2308984059']
 ['Íris María Birgisdóttir' '5469' '1603903879']
 ['Ívar Sigurðsson' '6324' '1309932659']
 ['Ágúst Guðni Ingason' '6558' '1601013180']
 ['Steinunn Guðlaug Gunnarsdóttir' '7923' '2006002580']
 ['Eydís Þorsteinsdóttir' '8003' '1703012420']
 ['Signý Guðrún Pálsdóttir' '8148' '2012012410']
 ['Elías Ari Heimisson' '9058' '0704012830']
 ['Mark Johnson' '9576' '0403983099']
 ['Víðir Kristjánsson' '9595' '2304003180']
 ['Jónas Valdimarsson' '9706' '0706012300']]
```

38C.3 Vartöluprófið kennitölurnar (í for-lykkju).

Vartöluprófið kennitölurnar (í for-lykkju).Vartöluprófið kennitölurnar (í for-lykkju). Ein þeirra stenst ekki prófun. Hvaða nemandi á hana?

In [7]:
```
import numpy as np
import requests
import io

def er_gild_kennitala(kennitala):
    vartala_sum = sum([int(str(kennitala)[i]) * [3, 2, 7, 6, 5, 4, 3, 2][i]
    afgangur = vartala_sum % 11
    if afgangur == 0:
        vartala = 0
    else:
        vartala = (11 - afgangur) % 10
        if vartala_sum == 110 and afgangur == 0:
            return False
    if vartala == 1:
        return False
    return vartala == int(kennitala[8])

# Load the data from the URL
url = "https://cs.hi.is/python/kennitolur.txt"
response = requests.get(url)
text = response.content.decode('utf-8')
data = np.genfromtxt(io.StringIO(text), delimiter=',', skip_header=1, dtype=

# Perform the vartölupróf on each kennitala using a for loop
invalid_count = 0
for i in range(data.shape[0]):
    kennitala = data[i, 2]
    if not er_gild_kennitala(kennitala):
        print(f"Nemandi {data[i, 0]} ({kennitala}) hefur ógilda kennitölu.")
        invalid_count += 1

# Print the count of invalid kennitalur
print(f"Number of invalid kennitalur: {invalid_count}")
```

```
Nemandi Signý Guðrún Pálsdóttir (2012012410) hefur ógilda kennitölu.
Nemandi Elías Ari Heimisson (0704012830) hefur ógilda kennitölu.
Number of invalid kennitalur: 2
```

Skil ekki afh ég fæ 2x ógildar kennitölur - SKOÐA SEINNA - bíða eftir svar frá kennara

38C.4 [Listi yfir mánuðir]

Hér er listi yfir mánaðarnöfn:

['janúar', 'febrúar', 'mars', 'apríl', 'maí', 'júní', 'júlí', 'ágúst', 'september', 'október', 'nóvember', 'desember'].

Skrifið fall sem tekur við kennitölu og skilar tilsvarandi fæðingardegi með sniði: „17. ágúst 1958“. Prófið með eigin kennitölu, og líka kennitölu einhvers í skránni sem fæddur er á annari öld.

In [15]:
```
def get_foedingardagur(kennitala):
    # Ná í fæðingadag frá kennitölu
    day = int(kennitala[0:2])
    month = int(kennitala[2:4])
```

```
    year_prefix = kennitala[4:6]
    century = None

    if year_prefix == "00":
        # Ef year_prefix er "00", varð fólk fætt á 20. öld
        century = 1900
    elif year_prefix <= "36":
        # Ef year_prefix er minna en eða jafnt og "36", varð fólk fætt á 20.
        century = 1900
    else:
        # Annars varð fólk fætt á 21. öld
        century = 2000
    year = century + int(year_prefix)

    # Ná í nafn mánaðar út frá mánaðarnúmeri
    months = ['janúar', 'febrúar', 'mars', 'apríl', 'maí', 'júní', 'júlí', '
    month_name = months[month - 1]

    # Skila strengnum
    return f"{day}. {month_name} {year}"
```

In [14]:
```
# Test með mína kennitölu
print(get_foedingardagur("2507943479"))

# Test með kennitölu frá 21.öld
print(get_foedingardagur("2112018000"))

# Test frá 20.öld
print(get_foedingardagur("2207863120"))
```

```
25. júlí 2094
21. desember 1901
22. júlí 2086
```

38D. Uppfléttitöflur

Lesið skrána sem notuðu var í dæmi C inn í þrjá lista eins og gert var þar.

In [21]:
```
import pandas as pd

url = "https://cs.hi.is/python/kennitolur.txt"

# Lesa gögn frá vefslóð í Dataframe
df = pd.read_csv(url, sep=",", header=0)

# Prenta DataFrame til að sjá töfluna
print(df)
```

	Nafn	Prófnúmer	Kennitala
0	Sigrún Jónsdóttir	176	2903993279
1	Kristín Fjóludóttir	542	1206972699
2	Birta Lárusdóttir	970	2605973109
3	Erla Ýr Guðnadóttir	1419	1210012330
4	Anton Ingi Þórsson	4854	2308984059
5	Íris María Birgisdóttir	5469	1603903879
6	Ívar Sigurðsson	6324	1309932659
7	Ágúst Guðni Ingason	6558	1601013180
8	Steinunn Guðlaug Gunnarsdóttir	7923	2006002580
9	Eydís Þorsteinsdóttir	8003	1703012420
10	Signý Guðrún Pálsdóttir	8148	2012012410
11	Elías Ari Heimisson	9058	704012830
12	Mark Johnson	9576	403983099
13	Víðir Kristjánsson	9595	2304003180
14	Jónas Valdimarsson	9706	706012300

38.1-2 Dictionary og raða í stafrófsröð

Búið til uppflettitöflu (dictionary) til að fletta upp á prófnúmeri út frá nafni (nafn er lykill og prófnúmer er gildi).

In [34]:
```
import pandas as pd

url = "https://cs.hi.is/python/kennitolur.txt"

# Lesa gögn frá vefslóð í pandas DataFrame
df = pd.read_csv(url, sep=",", header=0)

# Búa til dictionary
name_to_id = dict(zip(df['Nafn'], df['Prófnúmer']))

   # notað sem 'key' í sort eða sorted til að raða í íslenska stafrófsröð,
   # t.d. print(sorted(['ár','bára','bali','akur'], key=íslenska))"""
def isl_sort_key(s):
    isl_alphabet = list('aábcdðeéfghiíjklmnoópqrstuúvwxyýzþæö')
    return [isl_alphabet.index(c.lower()) for c in s if c.lower() in isl_alp

#  Raða dictionary með nafn í Íslensku stafrófi
sorted_names = sorted(name_to_id.keys(), key=isl_sort_key)

# Prenta töflu head
print("{:<30} {:<15}".format("NAFN", "PRÓFNÚMER"))

# prenta nafn og prófnúmer í stafrófsröð
for name in sorted_names:
    id_number = name_to_id[name]
    print("{:<30} {:<15}".format(name, id_number))
```

```
NAFN                          PRÓFNÚMER
Anton Ingi Þórsson            4854
Ágúst Guðni Ingason           6558
Birta Lárusdóttir             970
Elías Ari Heimisson           9058
Erla Ýr Guðnadóttir           1419
Eydís Þorsteinsdóttir         8003
Íris María Birgisdóttir       5469
Ívar Sigurðsson               6324
Jónas Valdimarsson            9706
Kristín Fjóludóttir           542
Mark Johnson                  9576
Signý Guðrún Pálsdóttir       8148
Sigrún Jónsdóttir             176
Steinunn Guðlaug Gunnarsdóttir 7923
Víðir Kristjánsson            9595
```

38D.3 [prímtölur númer 1 til 25]

Hér er listi yfir prímtölur númer 1 til 25:

P = [2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71,73,79,83,89,97]

Búið til uppflettitöflu, nrp, sem hægt er að nota til að fletta upp hvar gefin prímtala er í röðinni. Ef flett er upp á lyklinum P[i] á að fást gildið i+1 (ekki i af því Python byrjar að telja í 0). Notið for-lykkju sem rennir í gegn um P til að búa til uppflettitöfluna. Skrifið í framhaldi fall með tvo stika, tölu og nrp. Ef talan er stærri en 99 á fallið að skrifa að hún sé of stór, en annars á það að fletta upp í uppflettitöflunni og skrifa niðurstöðu uppflettingarinnar.

In [40]:
```python
P = [2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71,73,79,83,89,97]

# Búa til uppflettitöflu
nrp = {}
for i in range(len(P)):
    nrp[P[i]] = i + 1

def find_prime_number(tala, uppflettitoflun):
    if tala > 99:
        return f"{tala} er of stór"
    elif tala in uppflettitoflun:
        return f"{tala} er {uppflettitoflun[tala]}. prímtala"
    else:
        return f"{tala} er ekki prímtala"

# Prófa fallið með gefnum tölum
print(find_prime_number(13, nrp))
print(find_prime_number(16, nrp))
print(find_prime_number(103, nrp))
```

```
13 er 6. prímtala
16 er ekki prímtala
103 er of stór
```

VV3 Fall Rosenbrocks

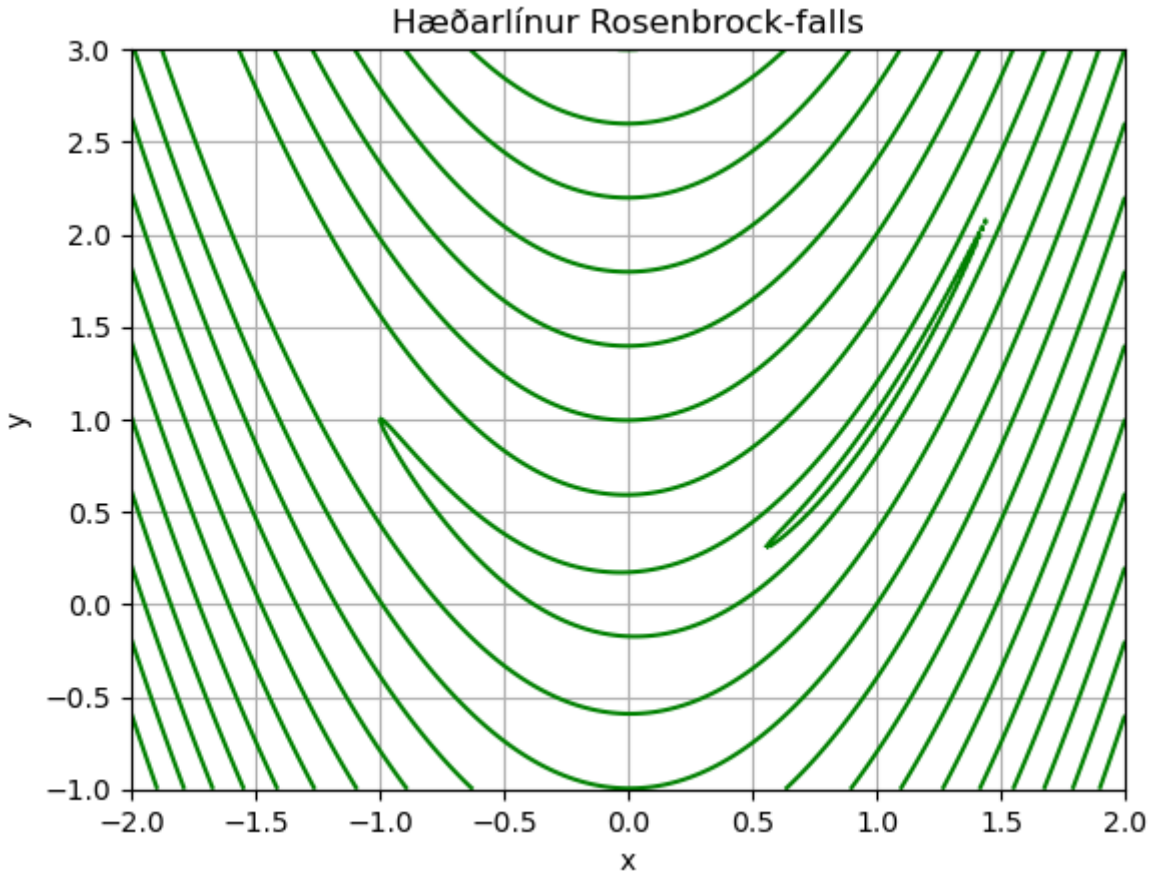
1.Skrifið fall rosen(x,y)

Skrifið fall Rosen (x,y) til að reikna gildi fallsins. Prófið að reikna $f(-1.2,1)$ sem ætti að gefa 24.2.

In [62]:
```
def rosenbrock(x, y):
    return (1 - x)**2 + 100 * (y - x**2)**2

x, y = -1.2, 1
result = rosenbrock(x, y)
rounded_result = round(result, 2)
print("Rosenbrock fall ({}, {}) is: {}".format(x, y, rounded_result))
```

```
Rosenbrock fall (-1.2, 1) is: 24.2
```

In [64]:
```
import numpy as np
import matplotlib.pyplot as plt

# Rosenbrock-fall
def rosenbrock(x, y):
    return (1 - x)**2 + 100 * (y - x**2)**2

# Búa til þétt net
x = np.linspace(-2, 2, 300)
y = np.linspace(-1, 3, 300)
X, Y = np.meshgrid(x, y)

# Reikna fallgildi í hverjum punkti
rosenbrock_vec = np.vectorize(rosenbrock)
Z = rosenbrock_vec(X, Y)

# Búa til hæðarlínur
levels = np.append([0.2], np.arange(2,50,4)**2)
plt.contour(X, Y, Z, levels=levels, colors='green')

# Bæta við netlínum
plt.grid(True)

# Setja titil og labele á ásana
plt.title('Hæðarlínur Rosenbrock-falls')
plt.xlabel('x')
plt.ylabel('y')

# Sýna myndina
plt.show()
```



VV6A (Stiglar og lágmörkun)

Gefið er fallið

f(x,y) = x^2y + 2xy^2 - 3xy + 4

punkturinn p=(1,1) og vigurinn u=(−1,−1)

VV6A.1 Ákvarðið stigul

Ákvarðið stigul f (þ.e.a.s.∇f), skrifið Python-fall sem finnur hann og notið það til að reikna ∇f(p)

In [66]:
```
import numpy as np

def gradient_f(x, y):
    df_dx = 2 * x * y + 2 * y**2 - 3 * y
    df_dy = x**2 + 4 * x * y - 3 * x
    return np.array([df_dx, df_dy])

p = np.array([1, 1])
gradient = gradient_f(p[0], p[1])
print("Stigull fallsins í punktinum ({}, {}) er: {}".format(p[0], p[1], grad
```

```
Stigull fallsins í punktinum (1, 1) er: [1 2]
```

VV6A.2

Lát u=(−1,−1)

- Notið Python-fallið úr a-lið líka til að reikna stefnuafleiðu f í stefnu u í punktinum p
  - Búið líka til einhvern punkt og vigur úr afmælisdegi ykkar og reiknið tilsvarandi stefnuafleiðu.

In [67]:
```
import numpy as np

def gradient_f(x, y):
    df_dx = 2 * x * y + 2 * y**2 - 3 * y
    df_dy = x**2 + 4 * x * y - 3 * x
    return np.array([df_dx, df_dy])

p = np.array([1, 1])
u = np.array([-1, -1])

grad_f = gradient_f(p[0], p[1])
directional_derivative = np.dot(grad_f, u)

print("Stefnuafleiðan í stefnu ({}, {}) í punktinum ({}, {}) er: {}".format(
```

```
Stefnuafleiðan í stefnu (-1, -1) í punktinum (1, 1) er: -3
```

In [70]:
```
import numpy as np

def gradient_f(x, y):
    df_dx = 2 * x * y + 2 * y**2 - 3 * y
    df_dy = x**2 + 4 * x * y - 3 * x
    return np.array([df_dx, df_dy])

point = np.array([25, 7, 113])
u = np.array([1, -1])

grad_f = gradient_f(point[0], point[1])
directional_derivative = np.dot(grad_f, u)

print("Stefnuafleiðan í stefnu ({}, {}) í punktinum ({}, {}, {} sem er afmæl
```

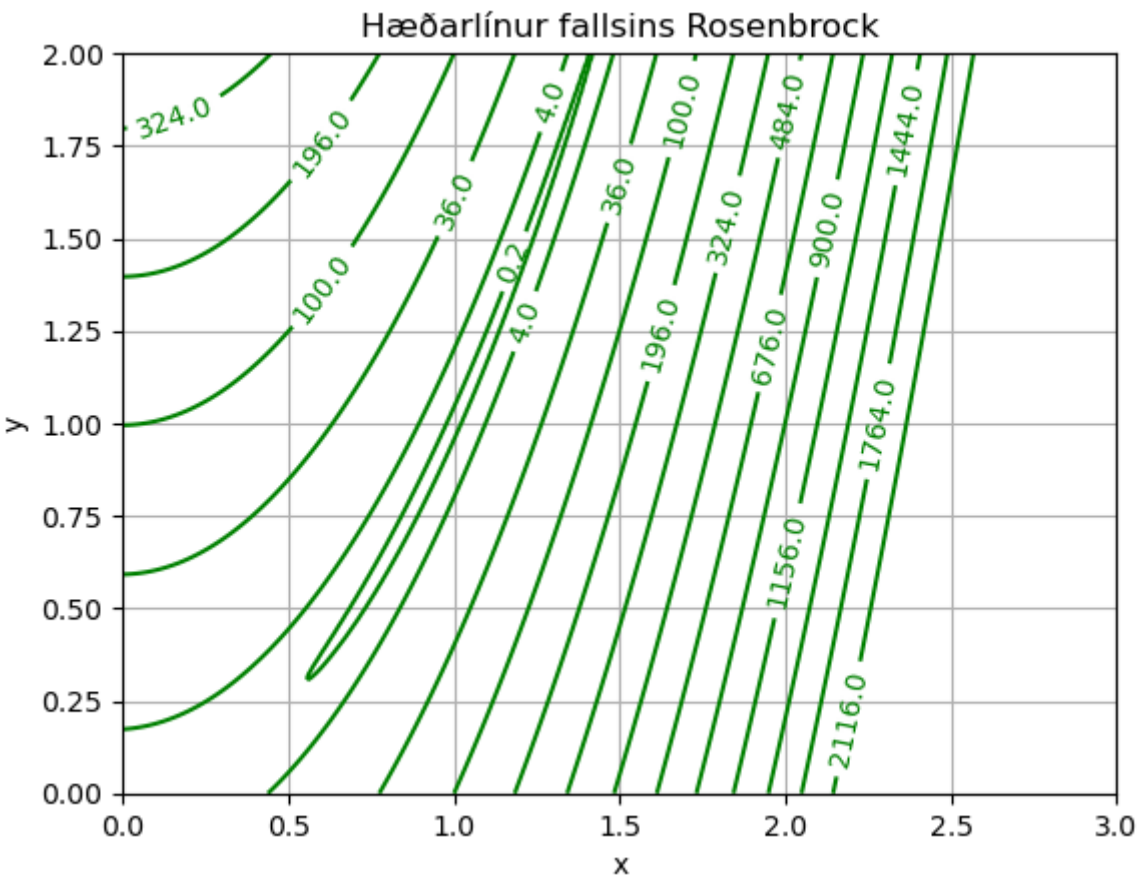
```
Stefnuafleiðan í stefnu (1, -1) í punktinum (25, 7, 113 sem er afmælið mitt
) er: -823
```

VV6A.4 Teikna

Teiknið að lokum hæðarlínur f á svæðinu [0,3]×[0,2]

In [6]:
```
import numpy as np
import matplotlib.pyplot as plt

def rosenbrock(x, y):
    return (1 - x)**2 + 100 * (y - x**2)**2

x = np.linspace(0, 3, 300)
y = np.linspace(0, 2, 200)
X, Y = np.meshgrid(x, y)
Z = rosenbrock(X, Y)

levels = np.append([0.2], np.arange(2, 50, 4)**2)

fig, ax = plt.subplots()
contour = ax.contour(X, Y, Z, levels=levels, colors='green')
ax.clabel(contour, fmt='%1.1f', fontsize=10)
ax.set_title('Hæðarlínur fallsins Rosenbrock')
ax.set_xlabel('x')
ax.set_ylabel('y')
```

```
ax.grid(True)

plt.show()
```



In [ ]: