Table of Contents

1. Što od navedenog je točno kada govorimo o računalnom razmišljanju? [više točnih odgovora]
    a. To je način rješavanja problema koji se može primijeniti na rješavanje problema iz života, ne samo problema povezanih s računarstvom.
    b. Računalno razmišljanje je misaoni proces tijekom kojeg definiramo problem te njegove manje dijelove na način da se rješenje može opisati kao slijed jednoznačno definiranih koraka.
    c. Način razmišljanja na matematički, odnosno računalni način kako bismo problem podijelili na manje dijelove i tako ga riješili.
    d. Sve navedeno.
2. Što je PEP?
    a. PEP je skraćenica od Python Education Professionals. To je certifikat koji imaju voditelji Python tečajeva.
    b. PEP je skraćenica od Programming Environment Python što predstavlja razvojno okruženje za programiranje u programskom jeziku Python.
    c. PEP je skraćenica od Python Enhancement Proposal. Svaki od tih prijedloga čini jedan dokument s podacima o smjeru i funkcionalnostima koje će se dodati u nove verzije Python programskog jezika.
    d. Ništa od navedenog.
3. Što su i zbog čega koristimo module u Pythonu?
    a. Modul je svaka .py datoteka u Pythonu. Koriste se za bolju organizaciju koda kako bi se kod lakše održavao (popravljale greške, dodavala proširenja).
    b. Moduli su zajednički naziv za dijelove koda kojeg ne želimo ponavljati. Recimo klase, funkcije i sl. Nije važno gdje se taj kod nalazi.
    c. Moduli su u stvari funkcionalnosti naše aplikacije. Recimo ispis podataka bio bi jedan modul.

d. Ništa od navedenog.

4. Klasa i objekti objasniti
   Klasa - korisnički definirani tip podatka i služi kao nacrt za kreiranje objekta. Klasa opisuje nešto iz stvarnog svijeta npr. životinja, račun, automobil.
   Objekt – stvarna instanca neke klase kojoj se dodjeljuje memorijsko prostor. Svaki objekt je definiran stanjem i ponašanjem definiranim u klasi.
5. Pandas objasniti (Panel dana)
   Pandas je alat za analiziranje podataka u Pythonu. Podatke sprema u DataFrame, a zanimljiv je jer u DataFrame možemo ubaciti podatke iz csv, sql, Excel i sl.
6. Odabrati tko je napravio python – Guido van Rossum u Nizozemskoj krajem 80-ih
7. Po cemu je dobio ime python – Monty Python's Flying Circus
8. Najpoznatija racunala za IOT – Raspberry PI, Arduino
9. Operacijski sustav za IOT – RasbianOS temeljen na Linoxu
10. Što je Numpy i je li numpy standardna biblioteka pythona.
    Numerican python nije dio standardne biblioteke pythona i koristi se za znanstvene proračune.
11. Razlika između Pandas i NumPy
    Pandas je Python biblioteka za obradu heterogenih tipova podataka, a NumPy je namijenjen za numeričke proračune pa je orijentiran na numeričke tipove podataka.
12. Kako se zove alat za upravljanje relacijskim bazama podataka koji se instalira zajedno s Python programskim jezikom? **SQLite**
13. Prepoznati grešku, a= 5/0 -> **Runtime error**
14. 3 pitanja sa vrstama erora, a nudi odgovore - sintaksa, runtime, ili logicki(bug) „a;=3“
15. Čemu služi Matplotlib
    Modul koji služi za vizualni prikaz podataka(npr. grafikon). Omogućava: grafove visoke kvalitete, ugradnju u GUI aplikacije, jednostavnu uporabu
16. Razlika između json.dump() i json.dumps() metoda? Kada koju koristimo?
    json.dump() koristimo kada želimo zapisati JSON direktno u JSON datoteku, a json.dumps() koristimo za generiranje JSON stringa (npr. za slanje putem mreže)
17. Dva osnovna tipa baze podataka koja se danas najčešće koriste
    Relacijske baze i NoSQL baze
18. Pet senzora uključenih u SenseHAT emulator

- Žiroskop
- Akcelerometar
- Magnetometar
- Temperatura
- Vlažnost
- Barometar

19. Iot je bio brzina i udaljenost
20. Vino klasika +density se trazio
21. Html. Parser je bio

Trebalo je duple izbaciti iz liste time da si imao dvije funkcije u kojoj jedna kad se poziva vraca true ili false ako se element ponavlja a druga funkcija brise element iz liste ako je dupli

```
def je_duplikat(lista, element):
    return lista.count(element) > 1

def ukloni_duplikate(lista):
    nova_lista = []
    for element in lista:
        if not je_duplikat(lista, element) or element not in nova_lista:
            nova_lista.append(element)
    return nova_lista

# Primjer korištenja
lista = [1, 2, 2, 3, 4, 4, 5]
rezultat = ukloni_duplikate(lista)
print(rezultat)  # Output: [1, 3, 5]
```

Napravi funkciju koja će izbrisati ponavljajući element iz liste i vratiti novu bez njih.

```
def ukloni_duplikate(lista):
    nova_lista = []
    for element in lista:
        if element not in nova_lista:
            nova_lista.append(element)
    return nova_lista

# Primjer korištenja
lista = [1, 2, 2, 3, 4, 4, 5]
rezultat = ukloni_duplikate(lista)
print(rezultat)  # Output: [1, 2, 3, 4, 5] Provjera niza
Provjera imena
```

Potrebno je analizirati dva podatkovna seta, bijela i crna vina:

- 'winequality-white.csv' i 'winequality-red.csv'.
- Pronađite parametar koji najviše korelira s gustoćom (density). Zatim iscrtajte histogram za crna i bijela vina s tim parametrom te zapišite kratak zaključak.
- Podatkovne setove možete preuzeti klikom na dugme "Preuzmi upute", a za rješenje zadatka potrebno je koristiti Visual Studio Code (Jupyter Notebook).
- Rješenje zadatka pohranite i prenesite na ispitnu platformu u formatu .zip datoteke.

```
import pandas as pd
import matplotlib.pyplot as plt

red_wine = pd.read_csv('winequality-red.csv', delimiter=';')
white_wine = pd.read_csv('winequality-white.csv', delimiter=';')

print(red_wine.head())
print(white_wine.head())
```

```
   fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
0            7.4              0.70         0.00             1.9      0.076
1            7.8              0.88         0.00             2.6      0.098
2            7.8              0.76         0.04             2.3      0.092
3           11.2              0.28         0.56             1.9      0.075
4            7.4              0.70         0.00             1.9      0.076

   free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  \
0                 11.0                  34.0   0.9978  3.51       0.56
1                 25.0                  67.0   0.9968  3.20       0.68
2                 15.0                  54.0   0.9970  3.26       0.65
3                 17.0                  60.0   0.9980  3.16       0.58
4                 11.0                  34.0   0.9978  3.51       0.56

   alcohol  quality
0      9.4        5
1      9.8        5
2      9.8        5
3      9.8        6
4      9.4        5
   fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
0            7.0              0.27         0.36            20.7      0.045
1            6.3              0.30         0.34             1.6      0.049
2            8.1              0.28         0.40             6.9      0.050
3            7.2              0.23         0.32             8.5      0.058
4            7.2              0.23         0.32             8.5      0.058

   free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  \
0                 45.0                 170.0   1.0010  3.00       0.45
1                 14.0                 132.0   0.9940  3.30       0.49
2                 30.0                  97.0   0.9951  3.26       0.44
3                 47.0                 186.0   0.9956  3.19       0.40
4                 47.0                 186.0   0.9956  3.19       0.40

   alcohol  quality
0      8.8        6
1      9.5        6
2     10.1        6
```

```
3     9.9       6
4     9.9       6
```

```
wine_data = pd.concat([red_wine, white_wine], ignore_index=True)
```

```
wine_data.replace([float('inf'), float('-inf')], float('nan'), inplace=True)
```

```
print(wine_data.head())
```

```
correlation_matrix = wine_data.corr()
```

```
print(correlation_matrix)
```

```
 fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
0          7.4             0.70         0.00             1.9      0.076
1          7.8             0.88         0.00             2.6      0.098
2          7.8             0.76         0.04             2.3      0.092
3         11.2             0.28         0.56             1.9      0.075
4          7.4             0.70         0.00             1.9      0.076

   free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  \
0                 11.0                  34.0   0.9978  3.51       0.56
1                 25.0                  67.0   0.9968  3.20       0.68
2                 15.0                  54.0   0.9970  3.26       0.65
3                 17.0                  60.0   0.9980  3.16       0.58
4                 11.0                  34.0   0.9978  3.51       0.56

   alcohol  quality
0      9.4        5
1      9.8        5
2      9.8        5
3      9.8        6
4      9.4        5
                      fixed acidity  volatile acidity  citric acid  \
fixed acidity              1.000000          0.219008     0.324436
volatile acidity           0.219008          1.000000    -0.377981
citric acid                0.324436         -0.377981     1.000000
residual sugar            -0.111981         -0.196011     0.142451
chlorides                  0.298195          0.377124     0.038998
free sulfur dioxide       -0.282735         -0.352557     0.133126
total sulfur dioxide      -0.329054         -0.414476     0.195242
```

density	0.458910	0.271296	0.096154
pH	-0.252700	0.261454	-0.329808
sulphates	0.299568	0.225984	0.056197
alcohol	-0.095452	-0.037640	-0.010493
quality	-0.076743	-0.265699	0.085532

	residual sugar	chlorides	free sulfur dioxide \
fixed acidity	-0.111981	0.298195	-0.282735
volatile acidity	-0.196011	0.377124	-0.352557
citric acid	0.142451	0.038998	0.133126
residual sugar	1.000000	-0.128940	0.402871
chlorides	-0.128940	1.000000	-0.195045
free sulfur dioxide	0.402871	-0.195045	1.000000
total sulfur dioxide	0.495482	-0.279630	0.720934
density	0.552517	0.362615	0.025717
pH	-0.267320	0.044708	-0.145854
sulphates	-0.185927	0.395593	-0.188457
alcohol	-0.359415	-0.256916	-0.179838
quality	-0.036980	-0.200666	0.055463

	total sulfur dioxide	density	pH	sulphates \
fixed acidity	-0.329054	0.458910	-0.252700	0.299568
volatile acidity	-0.414476	0.271296	0.261454	0.225984
citric acid	0.195242	0.096154	-0.329808	0.056197
residual sugar	0.495482	0.552517	-0.267320	-0.185927
chlorides	-0.279630	0.362615	0.044708	0.395593
free sulfur dioxide	0.720934	0.025717	-0.145854	-0.188457
total sulfur dioxide	1.000000	0.032395	-0.238413	-0.275727
density	0.032395	1.000000	0.011686	0.259478
pH	-0.238413	0.011686	1.000000	0.192123
sulphates	-0.275727	0.259478	0.192123	1.000000
alcohol	-0.265740	-0.686745	0.121248	-0.003029
quality	-0.041385	-0.305858	0.019506	0.038485

	alcohol	quality
fixed acidity	-0.095452	-0.076743
volatile acidity	-0.037640	-0.265699
citric acid	-0.010493	0.085532
residual sugar	-0.359415	-0.036980
chlorides	-0.256916	-0.200666
free sulfur dioxide	-0.179838	0.055463
total sulfur dioxide	-0.265740	-0.041385
density	-0.686745	-0.305858
pH	0.121248	0.019506
sulphates	-0.003029	0.038485
alcohol	1.000000	0.444319
quality	0.444319	1.000000

```
density = correlation_matrix['density'].drop('density')
most_correlated_attribute = density.idxmax()
print(f"Svojstvo koje najviše korelira sa vrijednošću ph je: {most_correlated_attribute}")


print(f"Vrijednost korelacija: {density[most_correlated_attribute]}")
```

```
Svojstvo koje najviše korelira sa vrijednošću ph je: residual sugar
Vrijednost korelacija: 0.5525169502932384
plt.figure(figsize=(10, 5))
plt.hist(wine_data['density'].dropna(), bins=30, edgecolor='k', alpha=0.7)
plt.title('Histogram za vrijednost gustoće')
plt.xlabel('density')
plt.ylabel('residual sugar')
plt.show()
```

FILM:

```
# TODO:
# Potrebno je implementirati konstruktor za razred Movie.
# Konstruktor prima dva argumenta, ime filma i godinu.
# Razred sadrzi dvije varijable instance (atribute): ime filma i godinu.
# Razred  Movie  morate iskoristiti u drugim podzadacima u ovom projektu.

import sqlite3
from datamodel.movie import Movie

class View:

    def __init__(self, name):
        self.name = name

    def movies(self):
        # TODO:
        # Potrebno je implementirati metodu `movies`.
        # Zadatak metode je spojiti se na bazu podataka imena `self.name`.
        # Metoda zatim mora iz baze dohvatiti sve filmove iz tablice `movies`.
        # Stupci u tablici se zovu `title` i `year`.
        # Metoda mora vratiti listu objekata klase `Movies`.
        # Primjer baze podataka nalazi se u `test_data/movies.db`
        # Baza podataka je Sqlite.
        movies = []
        conn = sqlite3.connect(self.name)
        cursor = conn.cursor()
```

```
        cursor.execute("SELECT title, year FROM movies")

        rows = cursor.fetchall()
        for row in rows:
            title, year = row
            movies.append(Movie(title, year))

        conn.close()

        return movies
class Movie:
    def _init_(self, title, year):
        self.title = title
        self.year = year
from datamodel.movie import Movie
import json


class Parser:

    def parse(self, file_name):
        # Parse JSON from file with file_name
        # Returns list of Movie objects
        movies = []
        with open(file_name, 'r') as file:
            # TODO:
            # Parsirajte JSON datoteku, pomocu  json  paketa.
            # Potrebno je iz liste filmova izvuci ime filma i godinu filma.
            # Primjer datoteke nalazi se u  test_data/movies.json
            # Metoda mora vracati listu objekata razreda  Movie .
            return movies
```

Palindrom

```
# Napišite funkciju koja provjerava da li je
# rijec (text) palindrom.
# Palindrom je rijec koja se cita isto s lijeva na desno,
# i s desna na lijevo (npr. "kisik").

# Funkcija se mora zvati  palindrom .
# Funkcija prima rijec kao argument koji mora biti tipa string (str).
# Funkcija vraća True ili False (bool).
# Ako je rijec palindrom funkcija vraca True, inace False.
# Ako ulazni parametar nije tipa str, funkcija mora vratiti False.
# Glavna funkcija (main) ispituje ispravnost rada funkcije,
# taj dio programskog koda ne treba mijenjati.
```

```
def palindrom(rijec):
    # TODO implementirajte funkciju
    pass


def main():
    assert palindrom('kisik')
    assert palindrom('a')
    assert palindrom('')
    assert palindrom('anavolimilovana')
    assert palindrom(5) is False
    assert palindrom('aa')
    assert palindrom('ovo nije palindrom') is False
    print("Implementacija je tocna!")


if _name_ == '_main_':
    main()
```

Računalno razmišljanje

```
# Primjenom računalnog razmišljanja osmislite i implementirajte
# funkciju koja traži postoji li element u sortiranoj listi.
# Funkcija vraća True ako je element u listi, inače False.
# Pretpostavka: lista je uvijek sortirana od najmanjeg do najvećeg elementa.

# Za pomoć imate tri funkcije:
#   -  sredina(lista)
#       - vraća srednji element u listi.
#   -  uzmi_desno_od_sredine(lista)
#       - vraća novu listu, od srednjeg elementa do kraja liste
#       - srednji element nije uključen u novu listu!
#   -  uzmi_lijevo_od_sredine(lista)
#       - vraća novu listu, od početka liste do srednjeg elementa
#       - srednji element nije uključen


def sredina(lista):
    mid = len(lista) // 2
    return lista[mid]


def uzmi_desno_od_sredine(lista):
    mid = len(lista) // 2
    return lista[mid + 1:]
```

```
def uzmi_lijevo_od_sredine(lista):
    mid = len(lista) // 2
    return lista[:mid]


def trazi(lista, element):
    # TODO: implementirajte ostatak funkcije!
    pass


# Program pri pokretanju testira ispravnost implementacije funkcije trazi
def main():
    lista = [1, 2, 3, 4, 5]
    for i in range(len(lista)):
        assert trazi(lista, lista[i])

    assert trazi([1, 2, 3, 4, 5], 10) is False

    lista = sorted(['a', 'c', 'z', 'f', 'e', 'e', 'r', 'w'])
    for i in range(len(lista)):
        assert trazi(lista, lista[i])

    assert trazi([], 10) is False

    print('Implementacija je ispravna')


if _name_ == '_main_':
    main()
```

sortiranje liste od najmanjeg do najveceg

```
def sortiraj(lista):
    lista = lista[:]
    for i in range(len(lista)):
        min_index = najmanji(lista, i, len(lista)-1)
        zamijeni(lista, i, min_index)
    return lista
```

Samoglasnici (brojanje)

```
Def broji_samoglasnike(text):

    text = str(text)
    samoglasnici = "aeiouAEIOU"
    broj_samoglasnika = 0
    for char in text:
```

```
        if char in samoglasnici:
            broj_samoglasnika += 1

    return broj_samoglasnika
```

Senzor

```
sensors = Sensors()

while True:
    temperature = sensors.get_temperature()

    if 18 <= temperature <= 24:
        print("IDEAL")
    elif temperature < 18:
        print("COLD")
    else:
        print("HOT")

    time.sleep(1)
```