1. **Din** cauza unei crize economice, s-a decis ca intr-o firma doar zilele de luni si marti sa fie zile de activitate, platite. Afisati (conform exemplului) pentru luna curenta, din anul curent, care sunt datele zilelor de activitate.

*Exemple:*

* pentru februarie 2022 se va afisa astfel:

*februarie 2022*: 1, 7, 8, 14, 15, 21, 22, 28

import datetime  
  
luni = {1: "ianuarie", 2: "februarie", 3: "martie", 4: "aprilie", 5: "mai", 6: "iunie", 7: "iulie", 8: "august",  
 9: "septembrie", 10: "octombrie", 11: "noiembrie", 12: "decembrie"}  
azi = datetime.date.today()  
azi = azi.replace(day=1)  
l = azi.month  
a = azi.year  
lista = []  
while azi.weekday() not in [0, 1]:  
 azi += datetime.timedelta(days=1)  
while azi.month == l:  
 if azi.weekday() in [0, 1]:  
 lista.append(azi.day)  
 azi += datetime.timedelta(days=1)  
 else:  
 azi += datetime.timedelta(days=1)  
a = str(a)  
print(luni[l], a+":", str(lista)[1:-1])

1. **Diana** si Elena sunt doua prietene bune ce au doar patru zile intre datele lor de nastere, Diana find mai mare (ex.: 11 mai si 15 mai, 30 mai si 3 iunie, etc.). Ele vor sa sarbatoreasca impreuna si, pentru a nu se supara, au stabilit sa aleaga ziua aflata la mijlocul distantei dintre cele doua zile de nastere. Cititi de la tastatura data de nastere a Dianei in formatul *zz.ll.aaaa* (nascuta prima), un numar natural *n* si cele *n* nume si prenume ale invitatilor. Folosind metoda *safe\_substitute*, creati pentru fiecare invitat un mesaj, ca in exemplu:

*Draga…(doar prenumele)…*

*Te invitam la petrecerea noastra, .. . (ziua din saptamana).... ..../data calendaristica...... la ora 18. Ne vom distra pe cinste!*

*Exemplu:*

Daca Diana este nascuta in data de 04.06.2001, suntem in anul 2022, iar unul dintre invitati se numeste Dumitrescu Laura, invitatia va fi:

Draga Laura,

Te invitam la petrecerea noastra, duminica, 6 februarie 2022, la ora 18. Ne vom distra pe cinste!

from string import Template  
import datetime  
import re  
  
data\_Dianei = input("Data de nastere a Dianei:")  
n = int(input("n ="))  
lista1 = []  
lista2 = []  
for nume in range(n):  
 lista1.append(input("Nume si prenume:"))  
for i in range(n):  
 lista2.append(re.split(" ", lista1[i]))  
for i in range(len(lista2)):  
 lista2[i].pop(0)  
lista2 = [prenume for lista in lista2 for prenume in lista]  
  
z = datetime.date.today()  
z = z.replace(month=int(data\_Dianei[3:5]), day=int(data\_Dianei[:2]))  
z += datetime.timedelta(days=2)  
zile = {0: "luni", 1: "marti", 2: "miercuri", 3: "joi", 4: "vineri", 5: "sambata", 6: "duminica"}  
luni = {1: "ianuarie", 2: "februarie", 3: "martie", 4: "aprilie", 5: "mai", 6: "iunie", 7: "iulie", 8: "august",  
 9: "septembrie", 10: "octombrie", 11: "noiembrie", 12: "decembrie"}  
  
  
t = Template("Draga $nume,\nTe invitam la petrecerea noastra, $ziua, $z $l $a, la ora 18. Ne vom distra"  
 " pe cinste!")  
for i in range(len(lista2)):  
 print(t.safe\_substitute(nume=lista2[i], ziua=zile[z.weekday()], z=z.day, l=luni[z.month], a=z.year))

1. **Se** citeste de la tastatura un text literar in limba romana, fara diacritice, care respecta tote regulile ortografice ale limbii romane (fara spatii nepotrivite, semne de punctuatie gresite, etc). Sa se afiseze ultimele doua cuvinte din fiecare fraza (se presupune cà fiecare fraza contine cel putin trei cuvinte). Consideram ca un cuvant poate contine cratima. Folositi ori de câte ori este posibil modulul *re*.

import re  
  
s = input("s =")  
s = re.split("[.!?]", s)

s = [x for x in s if x != ""]  
for fraza in s:  
 fraza = re.split("[, ]", fraza)  
 fraza = [i for i in fraza if i != ""]  
 print(fraza[-2:])

1. **Se** citeste n de la tastatura. Gigel, George si Gica genereaza aleator cate o lista de n numere de maxim 3 cifre. Castiga cel care are in lista mai putine numere palindrom. Se joaca 3 runde. Determinati cine este castigatorul dupa 3 runde (afisare in consola).

from random import randint  
  
  
def palindrom(a):  
 if a in range(0, 10):  
 return True  
 elif a in range(10, 100):  
 a = str(a)  
 if a[0] == a[1]:  
 return True  
 else:  
 return False  
 else:  
 a = str(a)  
 if a[0] == a[2]:  
 return True  
 else:  
 return False  
  
  
n = int(input("n ="))  
a = 0 # a=Gigel  
b = 0 # b=George  
c = 0 # c=Gica  
for k in range(3):  
 l1 = [] # 1=Gigel  
 l2 = [] # 2=George  
 l3 = [] # 3=Gica  
 for i in range(n):  
 l1.append(randint(0, 999))  
 l2.append(randint(0, 999))  
 l3.append(randint(0, 999))  
 for i in range(n):  
 if palindrom(l1[i]) is False:  
 a += 1  
 if palindrom(l2[i]) is False:  
 b += 1  
 if palindrom(l3[i]) is False:  
 c += 1  
  
if a == b == c:  
 print("Gigel, George si Gica sunt castigatorii!")  
elif max(a, b, c) == a == b:  
 print("Gigel si George sunt castigatorii!")  
elif max(a, b, c) == a == c:  
 print("Gigel si Gica sunt castigatorii!")  
elif max(a, b, c) == b == c:  
 print("George si Gica sunt castigatorii!")  
elif max(a, b, c) == a:  
 print("Gigel este castigatorul!")  
elif max(a, b, c) == b:  
 print("George este castigatorul!")  
elif max(a, b, c) == c:  
 print("Gica este castigatorul!")  
print(a, b, c)

1. **Din** fisierul *studenti.txt* se citesc, pe cate o linie, separate prin virgula si un singur spatiu, urmatoarele date despre un student:

* CNP;
* nume si prenume;
* un sir de note;

Scrieti o functie cu un numar variabil de argumente numite prin care sa se afiseze in fisierul *rezultate.out*, pentru fiecare student, pe cate o linie, numele, numarul de restante si media notelor pentru cei fara restante.

Exemplu:

**INPUT**

*studenti.txt*

6030303100600, Crisan Maria, 10, 9, 10, 9

5030303100870, Florea Dorin, 10, 9, 10, 8, 10

5021218123456, Manaila Alin, 7, 4, 9, 8, 6

6021027456456, Toader Georgiana, 4, 4, 8, 8, 6

**OUTPUT**

*rezultate.txt*

6030303100600, Crisan Maria, 0, 9.50

5030303100870, Florea Dorin, 0, 9.40

5021218123456, Manaila Alin, 1

6021027456456, Toader Georgiana, 2

import re  
  
  
def medii\_si\_restante():  
 f = open("studenti.txt")  
 f1 = open("rezultate.txt", "w")  
 s = re.split("\n", f.read())  
 CNP = []  
 nume = []  
 medii = []  
 restante = []  
 for i in range(len(s)):  
 s1 = re.split(", ", s[i])  
 CNP.append(s1[0])  
 s1.pop(0)  
 nume.append(s1[0])  
 s1.pop(0)  
 s1 = [int(x) for x in s1]  
 medii.append("{:.2f}".format(sum(s1)/len(s1)))  
 c = 0  
 for k in range(len(s1)):  
 if s1[k] < 5:  
 c += 1  
 restante.append(c)  
 for i in range(len(s)):  
 if restante[i] == 0:  
 f1.write(CNP[i]+", "+nume[i]+", "+"0, "+str(medii[i])+"\n")  
 else:  
 f1.write(CNP[i] + ", " + nume[i] + ", " + str(restante[i]) + "\n")  
  
  
medii\_si\_restante()

1. **Se** da un text literar in limba romana, fara diacritice, care nu respecta toate regulile ortografice ale limbii romane in care pot fi urmatoarele greseli: mai multe spatii consecutive, nu exista spatiu dupa semnele de punctuatie, exista spatiu inainte de semnele de punctuatie. Sa se afiseze frecventa de aparitie a fiecarui cuvant care incepe cu vocala (afisarea se va face conform exemplului). Folositi de cate ori este posibil modulul *re*.

*Exemplu*:

**Input:**

s=”Astazi este sambata! Avem examen la PP… Tu ai invatat? Ai lucrat laboratoarele?...”

**Output:**

Frecventa de aparitie a cuvantului “este” este 1.

Frecventa de aparitie a cuvantului “avem” este 1.

Frecventa de aparitie a cuvantului “examen” este 1.

Frecventa de aparitie a cuvantului “ai” este 2.

Frecventa de aparitie a cuvantului “invatat” este 1.

import re  
  
s = input("s =")  
s = s.lower()  
s = re.findall(r"\b[AEIOUaeiou]\S\*[a-zA-z]\b", s)  
l = [] # lista cu numarul de aparitie a fiecarui cuvant  
set = set() # set pentru a vedea cate cuvinte neduplicate sunt  
for i in range(len(s)):  
 c = 0 # contor  
 for cuv in s:  
 set.add(cuv)  
 if cuv == s[i]:  
 c += 1  
 l.append(c)  
for i in range(len(set)):  
 print("Frecventa de aparitite a cuvantului \"" + s[i] + "\" este " + str(l[i]))

1. **Din** fisierul *text\_literar.txt* se citeste un text pe mai multe linii (un numar par). Se afiseze un mesaj corespunzator: “Poezie” / “Proza”. Pentru ca textul sa fie poezie trebuie ca intre liniile din fisier sa existe una dintre rimele:

* imperecheata (rimeaza 1-2, 3-4, 5-6, etc.);
* incrucisata (rimeaza versurile 1-3, 2-4, 5-7, etc.);
* imbratisata (rimeaza versurile 1-4, 2-3, 5-8, 6-7, etc.);
* monorima (rimeaza versurile 1-2-3-4, 5-6-7-8, etc.).

Consideram ca doua versuri rimeaza daca au ultimele 3 caractere identice.

import re  
import string  
  
f = open("text\_literar.txt")  
s = re.split("\n", f.read())  
alfabet = list(string.ascii\_letters)  
s = [x.replace(".", "") for x in s]  
s = [x.replace(",", "") for x in s]  
s = [x.replace("?", "") for x in s]  
s = [x.replace("!", "") for x in s]  
s = [x for x in s if x != ""]  
  
  
def imperecheata():  
 for i in range(0, int(len(s)/4), 4):  
 if s[i][-3:] != s[i+1][-3:] or s[i+2][-3:] != s[i+3][-3:]:  
 return False  
 return True  
  
  
def incrucisata():  
 for i in range(0, int(len(s)/4), 4):  
 if s[i][-3:] != s[i+2][-3:] or s[i+1][-3:] != s[i+3][-3:]:  
 return False  
 return True  
  
  
def imbratisata():  
 for i in range(0, int(len(s)/4), 4):  
 if s[i][-3:] != s[i+3][-3:] or s[i+1][-3:] != s[i+2][-3:]:  
 return False  
 return True  
  
  
def monorima():  
 for i in range(0, int(len(s)/4), 4):  
 if s[i][-3:] != s[i+1][-3:] or s[i][-3:] != s[i+2][-3:] or s[i][-3:] != s[i+3][-3:] or \  
 s[i+1][-3:] != s[i+2][-3:] or s[i+1][-3:] != s[i+3][-3:] or s[i+2][-3:] != s[i+3][-3:]:  
 return False  
 return True  
  
  
if imperecheata() is True or incrucisata() is True or imbratisata() is True or monorima() is True:  
 print("Poezie")  
else:  
 print("Proza")

1. **Se** da un text literar in limba romana, fara diacritice, care respecta toate regulile ortografice ale limbii romane (fara spatii nepotrivite, semne de punctuatie gresite, etc.). Sa se afiseze in consola, pe o singura linie, separate prin virgula, cuvintele din text in ordinea crescatoare a numaruluiu de caractere, iar pentru cele de aceeasi lungime, ordonarea se va face alphabetic (vom considera majusculele inaintea literelor mici – conform ordinii codurilor ASCII).

*Exemplu*:

**Input**:

s=”Astazi este sambata ! Avem examen la PP… Tu ai invatat? Ai lucrat laboratoarele…”

**Output**:

Ai, Tu, PP, ai, la, Avem, este, Astazi, examen, lucrat, invatat, sambata, laboratoarele

import functools  
import re  
  
s = input("s =")  
s = re.split(r"[ ,.?!]", s)  
s = [x for x in s if x != ""]  
  
  
def comparatie(a, b):  
 if len(a) == len(b):  
 if a < b:  
 return -1  
 else:  
 return 1  
 if len(a) > len(b):  
 return 1  
 else:  
 return -1  
  
  
s.sort(key=functools.cmp\_to\_key(comparatie))  
print(\*s, sep=", ")

1. **Datele** de *25.12* si *26.12* din fiecare an sunt zile de concediu (indiferent de ziua din saptamana in care cad). Stiind ca intotdeauna sambata si duminica sunt zile libere, afisati pentru anul curent care va fi prima zi lucratoare de dupa Craciun (in format *zz.ll.aaaa*).

*Exemplu*: pentru anul *2020* s-ar afisa *28.12.2020*, iar pentru anul *2021* s-ar afisa *27.12.2021*

import datetime  
  
z = datetime.date.today()  
z = z.replace(month=12, day=26)  
z += datetime.timedelta(days=1)  
if z.weekday() == 5:  
 z += datetime.timedelta(days=2)  
elif z.weekday() == 6:  
 z += datetime.timedelta(days=1)  
print(str(z.day) + "." + str(z.month) + "." + str(z.year))

1. **Se** da un text literar in limba romana, fara diacritice, care respecta toate regulile ortografice ale limbii romane (fara spatii nepotrivite, semne de punctuatie gresite, etc.) Sa se afiseze primele doua cuvinte din fiecare fraza (se presupune ca fiecare fraza contine cel putin trei cuvinte). Folositi modulul ori de cate ori este posibil modulul *re*.

import re  
  
s = input("s =")  
s = re.split("[.!?]", s)  
s = [x for x in s if x != ""]  
for fraza in s:  
 fraza = re.split("[, ]", fraza)  
 fraza = [i for i in fraza if i != ""]  
 print(fraza[:2])

1. **Intr-o** firma s-a dovedit ca *work-from-home* a adus un randament mai mare de lucru, de aceea prima zi de luni din fiecare luna va fi declarata zi de concediu platit (bonus). Afisati (conform exemplului) care sunt aceste zile de concediu pentru anul in curs.

*Exemplu*: pentru anul 2021:

*4 ianuarie, 1 februarie, 1 martie, 5 aprilie, ...(etc)..., 6 decembrie*

import datetime  
  
luni = {0: "ianuarie", 1: "februarie", 2: "martie", 3: "aprilie", 4: "mai", 5: "iunie", 6: "iulie", 7: "august",  
 8: "septembrie", 9: "octombrie", 10: "noiembrie", 11: "decembrie"}  
z = datetime.date.today()  
z = z.replace(month=1, day=1)  
l = []  
for i in range(1, 13):  
 z = z.replace(month=i, day=1)  
 while z.weekday() != 0:  
 z += datetime.timedelta(days=1)  
 l.append(z.day)  
for k in range(len(l)):  
 print(l[k], luni[k], end=", ")

print("\b\b")

1. **Intr-o** companie, data de 1 mai este libera. Daca aceasta cade intr-o zi de joi, se considera ca ziua de vineri va fi, de asemenea, zi libera. Similar, daca 1 mai ar cadea intr-o zi de marti, atunci si ziua de luni va fi zi libera. Afisati care sunt toate zilele libere din preajma lui 1 mai anul curent, conform regulii anterioare.

*Exemple*: pentru anul 2019 s-ar afisa 1.05.2019 (1 mai 2019 a fost miercuri), pentru anul 2020 s-ar afisa 1.05.2020, 2.05.2020, 3.05.2020 (1 mai 2020 a fost vineri), pentru anul 2021 s-ar afisa 1.05.2021, 2.05.2021 (1 mai 2021 va fi sambata), iar pentru anul 2025 s-ar afisa 1.05.2025, 2.05.2025, 3.05.2025, 4.05.2025 (1mai 2025 cade joia).

import datetime  
  
z = datetime.date.today()  
z = z.replace(year=2025, month=5, day=1)  
l = []  
if z.weekday() == 0:  
 l.append(z - datetime.timedelta(days=2))  
 l.append(z - datetime.timedelta(days=1))  
 l.append(z)  
elif z.weekday() == 1:  
 l.append(z - datetime.timedelta(days=3))  
 l.append(z - datetime.timedelta(days=2))  
 l.append(z - datetime.timedelta(days=1))  
 l.append(z)  
elif z.weekday() == 2:  
 l.append(z)  
elif z.weekday() == 3:  
 l.append(z)  
 l.append(z + datetime.timedelta(days=1))  
 l.append(z + datetime.timedelta(days=2))  
 l.append(z + datetime.timedelta(days=3))  
elif z.weekday() == 4:  
 l.append(z)  
 l.append(z + datetime.timedelta(days=1))  
 l.append(z + datetime.timedelta(days=2))  
elif z.weekday() == 5:  
 l.append(z)  
 l.append(z + datetime.timedelta(days=1))  
elif z.weekday() == 6:  
 l.append(z - datetime.timedelta(days=1))  
 l.append(z)  
  
for i in range(len(l)):  
 print(str(l[i].day)+"."+str(l[i].month)+"."+str(l[i].year), end=", ")

print("\b\b")

1. **O** companie din Constanta are de livrat masini in alte **n** orase din tara si are drept scop principal ca suma timpilor de asteptare a clientilor sa fie minim. Din cauza unor defectiuni, compania are o singura platforma cu ajutorul careia pot fi livrate masinile. Cunoscand timpul necesar livrarii din orasul Constanta catre fiecare dintre cele **n** orase, determinati care este ordinea in care se va face livrarea, astfel incat timpul total de asteptare al clientilor sa fie minim. Se vor afisa in consola orasele, conform ordinii determinate. Daca exista mai multe posibilitati de a alege un oras catre care se va face mai intai livrarea, orasele vor fi preluate in ordine alfabetica.

Din fisierul *livrari.txt* se va citi, de pe cate o linie a fisierului, cate un oras si timpul de livrare catre respectivul oras (in ore).

*Exemplu:*

**INPUT**

livrari.txt

Cluj-Napoca 9

Brasov 9

Drobeta-Turnu Severin 8

Iasi 6

Timisoara 9

Arad 9

Satu-Mare 11

**OUTPUT**

Brasov, Iasi, Drobeta-Turnu Severin, Arad, Cluj-Napoca, Timisoara, Satu-Mare

import re  
import functools  
  
f = open("livrari.txt")  
s = re.split(r"\n", f.read())  
  
  
def comparatie(a, b):  
 distanta1 = re.split(r"\D", a)  
 distanta1 = [x for x in distanta1 if x != ""]  
 distanta2 = re.split(r"\D", b)  
 distanta2 = [x for x in distanta2 if x != ""]  
 if distanta1[0] > distanta2[0]:  
 return 1  
 elif distanta1[0] < distanta2[0]:  
 return -1  
 else:  
 if a[0] > b[0]:  
 return 1  
 else:  
 return -1  
  
  
s.sort(key=functools.cmp\_to\_key(comparatie))  
s.append(s.pop(s.index(s[0])))  
s = str(s)  
s = re.split(r"[\[\]',\d]", s)  
s = [x for x in s if x != ""]  
s = [x for x in s if x != " "]  
for i in range(len(s)):  
 print(s[i], end=", ")

print("\b\b")

1. **Se** da un text literar in limba romana, fara diacritice, care respecta toate regulile ortografice ale limbii romane (fara spatii nepotrivite, semne de punctuatie gresite, etc.). Sa se afiseze frazele cu acelasi numare de cuvinte, in ordinea crescatoare a numarului de cuvinte (afisarea se va face conform exemplului). Folositi de cate ori este posibil modulul *re.*

*Exemplu:*

**Input:**

Era ea... Poesis!... Cu toate astea nu putea sa fie ea. Paloarea mortala ca aceea a peretelui, parul cel blond, fata era a ei... Dar ea in acea pompa?... Ea, asa de saraca?... Dar intrasura a doua? Cine era?

**Output:**

*Propozitiile de lungime 2 sunt:*

Era ea...

Poesis!...

Cine era?

*Propozitiile de lungime 5 sunt:*

Ea, asa de saraca?...

Dar in trasura a doua?

*Propozitia de lungime 6 este:*

Dar ea in acea pompa?...

*Propozitia de lungime 8 este:*

Cu toate astea nu putea sa fie ea.

*Propozitia de lungime 13 este:*

Paloarea mortala ca aceea a peretelui, parul cel blond, fata era a ei...

import re  
  
s = input("s =")  
s = re.split("[.!?]", s)  
s = [x for x in s if x != ""]  
  
  
def cuvinte(a):  
 lista = re.split("[ ,]", a)  
 lista = [x for x in lista if x != ""]  
 return len(lista)  
  
  
s.sort(key=cuvinte)  
print(s)  
c = 0  
for i in s:  
 if cuvinte(i) > c:  
 print("Propozitiile de lungime "+str(cuvinte(i))+" sunt:")  
 c = cuvinte(i)  
 print(i)

1. **Se** da un text in limba romana, fara diacritice, care respecta toate regulile ortografice ale limbii romane (fara spatii nepotrivite, semne de punctuatie gresite, etc.). Sa se afiseze in consola, pe o singura linie, separate prin virgule, cuvintele din text ce contin cratime si se incheie cu o consoana. Se va folosi modulul *re*, utilizant un tipar pentru cuvintele ce indeplinesc conditia ceruta.

*Exemplu:*

**Input:**

s = "*Dintr-un nimic s-a transformat intr-o situatie intr-adevar grava.*"

**Output:**

*Dintr-un, intr-adevar*

import re  
  
s = input("s =")  
s = re.findall(r"[A-Za-z]+-[a-z]+", s)  
for cuv in s:  
 if cuv[-1] not in ["a", "e", "i", "o", "u"]:  
 print(cuv, end=", ")  
print("\b\b")

1. **Din** fisierul *sarbatori.txt* se citesc urmatoarele informatii (in aceasta forma):

01.01 Vasile, Vasilica

07.01 Ioan, Ion, Ioana

30.01 Grigore

10.02 Valentin, Valentina, Vali

23.04 Gheorghe, Georgeta, Gica, Geta

21.05 Elena, Constantin, Costel, Costin, Cosmin, Lenuta, Ilinca

29.06 Petru, Pavel, Petrica

12.07 Veronica

20.07 Ilie

30.08 Alex, Alexandru, Alexandra

08.09 Maria, Marian

08.11 Mihai, Gabriel, Gabi, Mihaita

06.12 Nicolae, Nicoleta, Nicu, Nicusor

07.12 Filofteia

27.12 Stefan, Stefania

O regula stabilita in M&K Company este ca, in oricare zi ce se regaseste in fisierul *sarabatori.txt* sa se acorde fiecarui angajat sarbatorit (ce are unul dintre prenume ce apare in fisier, in dreptul sarbatorii) un bonus echivalent cu 10% din salariul lunar.

Se citesc de la tastatura, de pe o singura linie, numele de familie, toate prenumele si salariul unui angajat, separate prin cate un spatiu. Sa se verifice daca astazi (data curenta) angajatul primeste sau nu bonus, astfel:

* in cazul in care ziua este una obisnuita (nu apare in fisierul *sarbatori.txt*) se va afisa mesajul Zi fara bonusuri;
* daca data curenta apare printre cele din fisier, atunci se va afisa mesajul: Angajatul … (nume prenume) primeste astazi un bonus de … (suma calculata, acel 10% din salariu) de lei.

*Exemplu:*

**INPUT:**

Georgescu Marian Grigore 4000

**OUTPUT:**

Daca data curenta este 04 feb. 2022, atunci se va afisa mesajul *Zi fara bonusuri*.

Daca data curenta este 30 ian. 2022, atunci se va afisa mesajul *Angajatul Georgescu Marina Grigore primeste astazi un bonus de 400 de lei.*

import datetime  
import re  
  
s = input("Nume: ")  
bonus = re.findall("\d+", s)  
nume = re.findall("\D+", s)  
f = open("sarbatori.txt")  
f = f.read()  
f = re.split("\n", f)  
f = [x for x in f if x != ""]  
  
Date = []  
Nume = []  
for i in f:  
 Date.append(re.findall("[\d.]+", i))  
 Nume.append(re.findall("[a-zA-z, ]+", i))  
Date = [x for lista in Date for x in lista]  
Nume = [s for lista in Nume for s in lista]  
  
z = datetime.date.today()  
for i in Date:  
 if int(i[:2]) == z.day and int(i[-2:]) == z.month:  
 index = Date.index(i)  
 lista\_nume = re.split("[, ]", Nume[index])  
 lista\_nume = [x for x in lista\_nume if x != ""]  
 nume\_angajat = re.findall("[A-Za-z]+", s)  
 for nume1 in lista\_nume:  
 for nume2 in nume\_angajat[1:]:  
 if nume1 == nume2:  
 print("Angajatul", nume[0], "primeste astazi un bonus de ", int(int(bonus[0])/10), "de lei.")  
 else:  
 print("Zi fara bonusuri.")  
 break  
 break  
 else:  
 print("Zi fara bonusuri.")  
 break

1. **Se** citeste din fisierul *text.txt* un text literar in limba romana, fara diacritice, care respecta toate regulile ortografice ale limbii romane (fara spatii nepotrivite, semne de punctuatie gresite, etc.). Fara a folosi structuri repetitive afisati in consola toate cuvintele ce contin exact 3 caractere (consideram ca un cuvant poate contine si cratima), ordonate alfabetic.

*Exemplu:*

**INPUT**

*text.txt*

Proiectul meu s-a incheiat azi, dar nu mai vrea sa lucrez cu voi.

**OUTPUT**

azi, dar, mai, meu, s-a, voi

import re  
  
s = input("s =")  
s = re.split("[ ,.!?]", s)  
s = [x for x in s if x != ""]  
l = list(filter(lambda x: len(x) == 3, s))  
l.sort()  
print(\*l, sep=", ")

1. **Scrieti** intr-o singura linie de cod (si fara a folosi structuri repetitive) pentru a afisa o singura data, in ordine lexicografica, vocalele folosite intr-un text citit din fisierul *text.txt*.

**INPUT:**  
Fisierul text.txt contine

*Ana are mere.*

**OUTPUT:**

*A, a, e*

import re; print(\*sorted(list(set(re.findall("[AERIOUaeiou]", open("text.txt").read())))), sep=", ")

1. **Fara** a folosi structuri repetitive (for, while) determinati suma codurilor ascii ale tuturor majusculelor, luate o singura data, dintr-un sir s citit de la tastatura.

*Exemplu:*

Pentru s = “Ana, Diana si Alina dau examen azi.” se va afisa 133.

import re; print(sum(list(map(ord, list(set(re.findall("[A-Z]", input("s ="))))))))

1. **Fara** a folosi structuri repetitive, determinati numarul de valori de 1 din scrierea binara a unui numar citit de la tastatura.

*Exemplu:*

pentru 147 se va afisa 4

import re  
  
n = int(input("n ="))  
n = re.findall("[1]", bin(n))  
print(sum(list(map(int, n))))

1. **Se** citeste de la tastatura un numar natural *n*, apoi *n* numere intregi ce se vor memora intr-o colectie A. Se citeste de la tastatura un numar natural *m*, apoi *m* numere intregi ce se vor memora intr-o colectie B.

*Exemplu:*

n = 6, A: 113, -4, 7, -4, -5, -109 si m = 7, B: 120, 178, -4, -4, -4, 120, -109

Sa se afiseze o singura data elementele negative din A care apar si in B.

*Exemplu:*

pentru colectiile de mai sus, se va afisa sirul -4, -109.

n = int(input("n ="))  
m = int(input("m ="))  
A = set()  
B = set()  
for i in range(n):  
 A.add(int(input("A =")))  
for j in range(m):  
 B.add(int(input("B =")))  
  
A = list((filter(lambda x: x < 0, A)))  
B = list((filter(lambda x: x < 0, B)))  
  
for k in range(len(A)):  
 for j in range(len(B)):  
 if A[k] == B[j]:  
 print(A[k], end=", ")  
print("\b\b")