MODALITATEA DE DESFĂȘURARE A EXAMENULUI LA DISCIPLINA "PROGRAMAREA ALGORITMILOR"

- Examenul la disciplina "Programarea algoritmilor" se va desfășura în ziua de 20.01.2022, între orele 9^{00} și 11^{30}, astfel:
  - 09^{00} – 09^{30}: efectuarea prezenței studenților
  - 09^{30} – 11^{30}: desfășurarea examenului
  - 11^{30} – 12^{00}: verificarea faptului că sursele trimise de către studenți au fost salvate pe platforma MS Teams
- Testul se va desfășura pe platforma MS Teams, iar pe tot parcursul desfășurării sale, de la ora 09^{00} la ora 12^{00}, studenții trebuie să fie conectați pe canalul dedicat cursului de "Programarea algoritmilor" corespunzător seriei lor.
- În momentul efectuării prezenței, fiecare student trebuie să aibă pornită camera video în MS Teams și să prezinte buletinul sau cartea de identitate. Dacă dorește să-și protejeze datele personale, studentul poate să acopere codul numeric personal și/sau adresa!
- În timpul desfășurării testului studenții pot să închidă camera video, dar trebuie să o deschidă dacă li se solicită acest lucru de către un cadru didactic!

- Toate subiectele se vor rezolva folosind limbajul Python.
- Subiectul 1 este obligatoriu, iar dintre subiectele 2, 3 și 4 se vor rezolva CEL MULT DOUĂ, la alegere.
- Citirea datelor de intrare se va realiza de la tastatură, iar rezultatele vor fi afișate pe ecran.
- Se garantează faptul că datele de intrare sunt corecte.
- Operațiile de sortare se vor efectua folosind funcții sau metode predefinite din limbajul Python.
- Pentru subiectul 1 nu contează complexitatea soluției propuse.
- Rezolvările subiectelor alese dintre subiectele 2, 3 și 4 trebuie să conțină:
  - o scurtă descriere a algoritmului și o argumentare a faptului că acesta se încadrează într-o anumită tehnică de programare;
  - în cazul problemelor rezolvate folosind metoda Greedy sau metoda programării dinamice se va argumenta corectitudinea criteriului de selecție sau a relațiilor de calcul;
  - în cazul subiectelor unde se precizează complexitatea maximă pe care trebuie să o aibă soluția, se va argumenta complexitatea soluției propuse și vor primi punctaj maxim doar soluțiile corecte care se încadrează în complexitatea cerută;
  - în cazul problemei rezolvate folosind metoda backtracking nu contează complexitatea soluției propuse, dar se va ține cont de eficiența condițiilor de continuare;
  - în fiecare program Python se va preciza, pe scurt, sub forma unor comentarii, semnificația variabilelor utilizate.
- Rezolvările corecte care nu respectă restricțiile indicate vor primi punctaje parțiale.
- Se acordă 1 punct din oficiu.

- Rezolvările tuturor subiectelor se vor scrie de mână, folosind pix/stilou cu culoarea pastei/cernelii albastră sau neagră. Pe fiecare pagina studentul își va scrie numele și grupa, iar paginile trebuie să fie numerotate.
- Înainte de expirarea timpului alocat examenului, toate paginile vor fi fotografiate/scanate clar, în ordinea corectă, și transformate într-un singur fișier PDF care va fi încărcat în Google Drive folosind un anumit formular.

- Numele fișierului PDF trebuie să respecte șablonul *grupa_nume_prenume.pdf*. De exemplu, un student cu numele Popescu Ion Mihai din grupa 131 trebuie să denumească fișierul care conține rezolvările tuturor subiectelor astfel: *131_Popescu_Ion_Mihai.pdf*.

Subiectul 1 – limbajul Python – 3 p.

a) Scrieţi o funcţie **`litere`** care primeşte un număr variabil de cuvinte formate din litere mici ale alfabetului englez şi returnează un dicţionar care conţine pentru fiecare cuvânt primit ca parametru, un dicţionar cu frecvenţa fiecărei litere distincte care apare în cuvânt. De exemplu, pentru apelul **`litere('teste', 'dictionar', 'ele')`** funcţia trebuie să returneze dicţionarul `{'teste': {'e': 2, 's': 1, 't': 2}, 'dictionar': {'a': 1, 'c': 1, 'd': 1, 'i': 2, 'n': 1, 'o': 1, 'r': 1, 't': 1}, 'ele': {'e': 2, 'l': 1}}`. **(1.5 p.)**

b) Folosind un dicţionar cu acelaşi format ca valorile dicţionarului de la punctul a) (i.e., cheile sunt litere, iar valorile frecvenţa literei respective), să se scrie o secvenţă de iniţializare (*list comprehension*) pentru o listă astfel încât aceasta să conţină perechile de forma **`(literă, frecvență)`** cu literele extrase din dicţionar care au frecvenţa pară. De exemplu, pentru dicţionarul `{'e': 2, 's': 1, 't': 2}` lista trebuie să fie `[('e', 2), ('t', 2)]`. **(0.5 p.)**

c) Considerăm următoarea funcţie recursivă:

```
def f(lista, p, u):
    if u-p <= 1:
        return sum(lista[p: u+1])
    k = (u-p+1) // 3
    aux_1 = f(lista, p, p+k)
    aux_2 = f(lista, p+k+1, p+2*k)
    aux_3 = f(lista, p+2*k+1, u)
    return aux_1 + aux_2 + aux_3
```

Determinaţi complexitatea funcţiei apelată pentru o listă **`L`** formată din **`n`** numere întregi astfel: **`f(L, 0, n-1)`**. **(1 p.)**

Subiectul 2 – metoda Greedy (3 p.)
Complexitatea maximă a soluției: O(n $\log_2$ n)

Bunicul vostru are N recipiente care conţin diverse cantităţi de apă la diferite temperaturi, respectiv recipientul i conține X_i litri de apă la temperatura Y_i. Bunicul vostru dorește să facă baie şi crede cu tărie faptul că apa în care se va spăla trebuie să aibă exact o temperatură T. De asemenea, bunicul consideră că este cu atât mai bine cu cât are mai multă apă la dispoziție. Din păcate, pe bunic nu îl ține spatele, așa că vă roagă pe voi să alegeţi inteligent ce cantitate de apă din fiecare recipient trebuie să turnaţi în cadă astfel încât bunicul vostru să se poată bucura de o baie plăcută! Temperatura apei nu este afectată de mediul exterior. La final, dacă aţi turnat $0 \leq C_i \leq X_i$ din recipiente, temperatura finala a apei din cadă va fi egală cu:

$$F = \frac{C_1 * Y_1 + C_2 * Y_2 + \cdots + C_N * Y_N}{C_1 + C_2 + \cdots + C_N}$$

Datele de intrare se vor citi de la tastatură, astfel:

- de pe prima linie două numere separate printr-un spațiu, respectiv numărul natural strict pozitiv N de recipiente şi numărul real T reprezentând temperatura dorită de bunicul vostru pentru apa din cadă;
- de pe următoarele N linii se vor citi câte două numere reale X_i şi Y_i cu semnificatia din enunţ (cantitatea de apă din recipientul i şi temperatura sa).

Programul Python trebuie să afişeze pe ecran cele N valori reale C_i, fiecare pe câte o linie, cu exact 3 zecimale. Se acceptă orice soluţie corectă.

Exemplu:

Date de intrare	Date de ieșire
3 40 50 20 72.5 50 38.72 90	50.000 72.500 5.500

Explicaţii: În cadă s-au turnat exact 50.000 + 72.5 + 5.5 = 128 litri, iar temperatura finală a apei este (50.000 * 20 + 72.5 * 50 + 5.5 * 90) / 128 = 5120 / 128 = 40 de grade. Nu există nicio altă variantă prin care să se toarne în cadă mai mult de 128 de litri şi apa să aibă exact temperatura de 40 de grade!

Subiectul 3 – metoda Programării Dinamice (3 p.)
Complexitatea maximă a soluției: $O(n^2)$

După o nouă plimbare lungă prin parc cu stăpâna sa, cățelușa Laika se află în fața unei mari provocări: trebuie iar să urce cele n trepte până la ușa apartamentului în care locuiește. De data aceasta ea mai are însă energie și poate să sară de pe o treaptă i direct pe una dintre treptele i+1, i+2, ..., n . Totuși pentru a sari *i* trepte trebuie sa plătească un cost c_i , pentru că face gălăgie. De asemenea, pentru că treptele au fost spălate și Laika vine după o joacă prin noroi, pentru fiecare treaptă *i* (i=1,...,n) pe care calcă Laika există un cost t_i (număr natural pozitiv) pe care trebuie să îl plătească. Ajutați-o pe Laika scriind un program Python care afișează o modalitate de a urca treptele de la baza scării (considerată treapta 0, pentru care taxa este $t_0 = 0$) la treapta *n* cu cost total minim. Se citesc de la tastatură n și taxele pentru cele n trepte $t_1, t_2, ..., t_n$ și costul pentru a sări treptele $c_1, c_2, ..., c_n$ (c_i este costul pe care trebuie să îl plătească dacă sare *i* trepte). Laika nu e un câine normal așa că s-ar putea costul să sară 3 trepte să fie mai mic decât să sară 2.

Intrare de la tastatură	Ieșire pe ecran
5 1 2 11 1 1 1 7 4 9 5	taxa totala 6 pentru traseul cu scările 0 5

Explicații: n=5, deci scara are 5 trepte numerotate 1,2, ... ,5 (pe lângă baza scării care se consideră treapta 0 și pentru care nu se plătește taxă); Laika preferă să sară din prima 5 trepte pentru că poate :), deci costul plătit va fi $c_5 + t_5 = 5 + 1 = 6$ (c_5 pentru că a sărit 5 trepte și t_5 pentru că a călcat pe treapta 5)

Intrare de la tastatură	Ieșire pe ecran
5 1 2 11 1 1 1 7 4 9 15	taxa totala 9 pentru traseul cu scările 0 1 4 5

Explicații: Laika sare la treapta 1 cu costul 2 (c_1=1 costul de a sări o treaptă, t_1=1 costul de a ajunge pe treapta 1), apoi Laika sare la treapta 4 cu costul 5 (c_3=4 costul de a sări 3 trepte, t_4=1 costul de călca pe treapta 4), apoi sare la treapta 5 cu costul 2 (1+1).

Subiectul 4 – metoda Backtracking (3 p.)

a) Un număr natural se numeşte *echilibrat* dacă valoarea absolută a diferenţei dintre oricare două cifre aflate pe poziţii consecutive şi, respectiv, între prima şi ultima sa cifră este 0 sau 1. De exemplu, numerele 5456 şi 1232 sunt echilibrate, iar numerele 5443, 5446 şi 1233 nu sunt echilibrate. Scrieţi un program Python care citeşte de la tastatură un număr natural m ($2 \leq m \leq 20$) şi afişează toate numerele naturale *echilibrate* formate din exact m cifre, precum şi numărul acestora. **(2.5 p.)**

Exemplu:
Pentru $m = 4$ trebuie afişate următoarele numere (nu neapărat în această ordine):

1000	3232	5456	7767
1001	3233	5544	7776
1010	3234	5545	7777
1011	3322	5554	7778
1012	3323	5555	7787
1100	3332	5556	7788
1101	3333	5565	7876
1110	3334	5566	7877
1111	3343	5654	7878
1112	3344	5655	7887
1121	3432	5656	7888
1122	3433	5665	7898
1210	3434	5666	8767
1211	3443	5676	8777
1212	3444	6545	8778
1221	3454	6555	8787
1222	4323	6556	8788
1232	4333	6565	8789
2101	4334	6566	8877
2111	4343	6567	8878
2112	4344	6655	8887
2121	4345	6656	8888
2122	4433	6665	8889
2123	4434	6666	8898
2211	4443	6667	8899
2212	4444	6676	8987
2221	4445	6677	8988
2222	4454	6765	8989
2223	4455	6766	8998
2232	4543	6767	8999
2233	4544	6776	9878
2321	4545	6777	9888
2322	4554	6787	9889
2323	4555	7656	9898
2332	4565	7666	9899
2333	5434	7667	9988
2343	5444	7676	9989
3212	5445	7677	9998
3222	5454	7678	9999
3223	5455	7766	Nr. solutii:159

b) Precizaţi cum ar trebui modificată o singură instrucţiune din program astfel încât să fie afişate doar numerele *echilibrate* formate din m cifre şi care sunt numere pare. Pentru exemplul anterior, aceste soluţii sunt cele scrise cu roşu. **(0.5 p.)**