MODALITATEA DE DESFĂȘURARE A EXAMENULUI LA DISCIPLINA "PROGRAMAREA ALGORITMILOR" DIN SESIUNEA DE RESTANȚE 31.05.2022 – 06.06.2022

- Examenul la disciplina "Programarea algoritmilor" se va desfășura în ziua de 03.06.2022, între orele 9^{00} și 14^{00}, astfel:
  - 09^{00} – 9^{15}: efectuarea prezenței studenților la testul de laborator
  - 09^{15} – 10^{45}: desfășurarea testului de laborator
  - 10^{45} – 11^{00}: verificarea faptului că sursele trimise de către studenți au fost salvate pe platformă
  - 11^{00} – 11^{30}: pauza
  - 11^{30}– 11^{45}: efectuarea prezenței studenților la examenul scris
  - 11^{45} – 13^{45}: desfășurarea examenului scris
  - 13^{45} – 14^{00}: verificarea faptului că fișierele trimise de către studenți au fost salvate pe platforma

- Ambele probe se vor desfășura pe platforma MS Teams, iar pe tot parcursul desfășurării lor studenții trebuie să fie conectați pe canalul dedicat cursului de "Programarea algoritmilor" corespunzător seriei lor (pe întâlnirea special creata pentru examen).
- 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!

Precizări privind desfășurarea testului de laborator:

- Testul va conține 3 subiecte, iar un subiect poate să aibă mai multe cerințe.
- Rezolvarea unui subiect se va realiza într-un singur fișier sursă Python (.py), indiferent de numărul de cerințe, care va fi încărcat/atașat ca răspuns pentru subiectul respectiv.
- Numele fișierului sursă Python trebuie să respecte următorul șablon: *grupa_nume_prenume_subiect.py*. De exemplu, un student cu numele Popescu Ion Mihai din grupa 131 trebuie să denumească fișierul care conține rezolvarea primului subiect astfel: 131_Popescu_Ion_Mihai_1.py.
- La începutul fiecărui fișier sursă Python se vor scrie, sub forma unor comentarii, numele și prenumele studentului, precum și grupa sa. Dacă un student nu reușește să rezolve deloc un anumit subiect, totuși va trebui să încarce/atașeze un fișier sursă Python cu informațiile menționate anterior!
- Toate rezolvările (fișierele sursă Python) trimise de către studenți vor fi verificate din punct de vedere al similarității folosind un software specializat, iar eventualele fraude vor fi sancționate conform Regulamentului de etică și profesionalism al FMI (http://old.fmi.unibuc.ro/ro/pdf/2015/consiliu/Regulament_etica_FMI.pdf).
- Operațiile de sortare se vor efectua folosind funcții sau metode predefinite din limbajul Python.

Precizări privind desfășurarea examenului scris:

- 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.
- 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 fiecare program Python se va preciza, pe scurt, sub forma unor comentarii, semnificația variabilelor utilizate.
- Pentru subiectele 1 nu contează complexitățile soluțiilor propuse.
- 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 pe platforma MS Teams 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 rezolvarea primului subiect 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 doar din litere mici ale alfabetului englez şi returnează informaţii despre aceste cuvinte sub forma unui dicţionar de perechi {**cuvant**: **lista de litere**}. Cheia este un cuvânt primit ca parametru, iar **lista de litere** este formată din literele care au mai mult de o apariţie în cuvânt şi prima apariţie este pe o poziţie pară (numerotarea poziţiilor începe de la 0). Listele de litere nu conţin elemente duplicate şi sunt sortate în ordine lexicografică. De exemplu, pentru apelul **litere**("**neintentionatei**", "**succesiuni**", "**cucurbitaceelor**", "**dicotiledonat**") funcţia va returna dicţionarul `{'neintentionatei': ['i', 'n', 't'], 'succesiuni': ['c', 'i', 's'], 'cucurbitaceelor': ['c', 'e', 'r'], 'dicotiledonat': ['d', 't']}`. **(1.5 p.)**

b) Înlocuiţi punctele de suspensie din instrucţiunea **`numere = [...]`** cu o secvenţă de iniţializare (list comprehension) astfel încât, după executarea sa, lista să conţină numerele naturale formate din exact două cifre nenule distincte şi în care cifrele nu se divid între ele. **(0.5 p.)**

c) Considerăm următoarea funcţie recursivă:

```
def f(lista):
    if len(lista) <= 2:
        return max(lista)
    k = len(lista) // 2
    aux_1 = lista[:k]
    aux_2 = lista[k:]
    return max(f(aux_1), f(aux_2))
```

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

Subiectul 2 – metoda Greedy (3 p.)

Complexitatea maximă a soluției: $\mathcal{O}(mn + n \log_2 n)$

Studiosul elev Algebrel tocmai a învățat la școală despre matrice. Curios din fire, el ar vrea să afle care este produsul maxim pe care ar putea să-l obțină selectând anumite linii din matrice (cel puțin una și posibil toate!) și înmulțind toate elementele de pe liniile respective. Scrieți un program Python care citește de la tastatură două numere naturale nenule *n* și *m*, o matrice cu *n* linii și *m* coloane, după care afișează pe ecran, în forma indicată în exemplu, indicii liniilor pe care trebuie să le selecteze Algebrel (nu contează în ce ordine), precum și produsul maxim pe care el poate să-l obțină înmulțind toate elementele de pe liniile respective.

Exemplu:

Date de intrare	Date de ieșire
5 4 5 7 -1 -2 8 -1 -5 -3 1 -1 3 6 3 3 3 2 5 -7 1 2	0 1 3 4 31752000

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

Hoțomel este un hoț de nasturi. El a văzut într-un magazin puși în rând unul lângă altul n săculețe cu nasturi, numerotați de la stânga la dreapta 1, 2, ..., n. Pe fiecare săculeț este trecut numărul de nasturi din el. Hoțomel nu vrea să fure toate săculețele, de teamă să nu fie prins (și pentru că ar vrea să mai lase, totuși, niște nasturi în magazin), așa că se gândește ca din două săculețe alăturate să ia cel mult unul, astfel încât să nu fie spații goale mari în șirul de săculețe rămase. Hoțomel nu știe însă programare dinamică (lui îi plac nasturii, nu programarea), așa că vă roagă pe voi să scrieți un program Python care citește de la tastatură numărul n de săculețe și numerele p_1, p_2,..., p_n reprezentând numărul de nasturi din săculețele 1,2, ..., n și care afișează ce săculețe să ia Hoțomel astfel încât numărul total de nasturi furați să fie maxim, respectând restricția că nu vrea să ia două săculețe alăturate (cu indicii i și i+1). Hoțomel promite că dacă rezolvați corect problema nu va fura nasturii, ci îi va cumpăra, dar ar vrea totuși să știe cât de mulți ar putea lua cu strategia lui.

Intrare de la tastatură	Ieșire pe ecran
7 2 9 11 10 1 1 6	Saculetii 2 4 7 25 nasturi

Explicații: Hoțomel va lua săculețele cu numerele 2, 4 și 7 și va obține 9 + 10 + 6 = 25 de nasturi (nu ar fi putut lua și săculețul 3 și săculețul 4, pentru că sunt alăturați!).

Subiectul 4 – metoda Backtracking (3 p.)

a) Lăcomel are un rucsac de capacitate *G* şi are la dispoziţie *n* obiecte numerotate de la 1 la *n*. Pentru fiecare obiect el ştie greutatea şi câştigul pe care l-ar obţine la încărcarea lui în rucsac. Lăcomel ar vrea să încarce obiecte în rucsac (fără a le fracţiona!) astfel încât câştigul total obţinut (suma câştigurilor obiectelor încărcate) să fie mulţumitor pentru el, adică să fie cuprins în intervalul [*a*, *b*]. El vă roagă să scrieţi un program Python care să citească de la tastatură numerele *G*, *n*, greutăţile celor n obiecte, câştigurile celor *n* obiecte şi numerele *a* şi *b* (toate datele sunt numere naturale), după care să afişeze toate modalităţile de a încărca obiecte în rucsac fără a depăşi capacitatea *G* astfel încât câştigul total obţinut să fie în intervalul [*a*, *b*] (se vor afişa indicii obiectelor care trebuie încărcate) **(2.5 p.)**

Exemplu: Pentru *G* = 8, *n* = 6, greutăţile 1, 2, 2, 4, 3, 3, câştigurile 1, 2, 5, 4, 4, 2, *a* = 6 şi *b* = 9 trebuie afişate următoarele variante de încărcare a obiectelor în rucsac (nu neapărat în această ordine):

5 6
4 6
4 5
3 6
3 5
3 4
2 5
2 5 6
2 4
2 3
2 3 6
1 5 6
1 4 6
1 4 5
1 3
1 3 6
1 2 5
1 2 4
1 2 3

b) Modificaţi o singură instrucţiune din program astfel încât să fie afişate doar soluţiile care conţin cel putin un obiect dintre cele cu câştig maxim. Pentru exemplul anterior, aceste soluţii sunt cele scrise cu roşu. **(0.5 p.)**