Documetație

Problema 1 Numere complexe:

Creați un program care lucrează cu numere complexe (a + bi). Programul

gestionează o listă de numere complexe și permite efectuarea repetată a

următoarelor acțiuni:

1. Adaugă număr în listă.

• Adaugă număr complex la sfârșitul listei

• Inserare număr complex pe o poziție dată.

2. Modifică elemente din listă.

• Șterge element de pe o poziție dată.

• Șterge elementele de pe un interval de poziții.

• Înlocuiește toate aparițiile unui număr complex cu un alt număr

complex.

3. Căutare numere.

• Tipărește partea imaginara pentru numerele din listă. Se dă intervalul

de poziții (sub secvența).

• Tipărește toate numerele complexe care au modulul mai mic decât 10

• Tipărește toate numerele complexe care au modulul egal cu 10

4. Operații cu numerele din listă

• suma numerelor dintr-o subsecventă dată (se da poziția de început și

sfârșit).

• Produsul numerelor dintr-o subsecventă dată (se da poziția de început

și sfârșit).

• Tipărește lista sortată descrescător după partea imaginara

5. Filtrare.

• Filtrare parte reala prim – elimină din listă numerele complexe la care

partea reala este prim.

• Filtrare modul – elimina din lista numerele complexe la care modulul

este <,= sau > decât un număr dat.

6. Undo

• Reface ultima operație (lista de numere revine la numerele ce existau

înainte de ultima operație care a modificat lista)

Plan iterații:

-Iterația 1:

terminare funcționalitățile 1,2,3(varianta procedurală) + documentație

-Iterația 2:

terminare funcționalități 4,5,6 (proiect final variantă modulară) + documentație finală

-Iterația 3:

optimizări extra pentru posibile probleme

Notă: Toate numere complexe vor fi stocate intr-o listă numită numere\_complexe, inițial vidă, declarată la începutul programului

Funcții folosite general

|  |  |
| --- | --- |
| Denumire | Scop |
| alegere\_optiune() | Folosit pentru a nagiva interfata |
| citire\_nr\_complex() | Folosit pentru citirea unui numar complex de la tastura |
| citire\_index() | Folosit pentru a putea accesa secvente sau anumite pozitii dintr-un sir |

1. **Adaugă număr în listă:**

Funcții

|  |  |
| --- | --- |
| Denumire | Descriere |
| add\_la\_final(real, imag, lista) | Adauga la finalul listei date prin parametru un numar complex format din parametrii real si imag |
| add\_cu\_index(real, imag, index, lista) | Inserează pe pozitie data prin parametrul index in lista data prin parametru un numar complex format din parametrii real si imag |

Scenariu de rulare pentru adăugarea numărului 13+14i la finalul listei:

* Afisare meniu principal
* p = 1 (user input prin alegere\_optiune() )
* afisare meniu functionalitate 1
* p = 1(user input prin alegere\_optiune() )
* apelare citire\_nr\_complex()
* input\_tuple = (13,14) din citire\_nr\_complex()
* apelare add\_la\_final(input\_tuple[0], input\_tuple[1], numere\_complexe)
* la finalul listei numere\_complexe va fi adaugat numarul (13+14i)

Scenariu de rulare pentru adăugarea numărului 13+14i pe poziția 3:

* Afisare meniu principal
* p = 1 (user input prin alegere\_optiune() )
* afisare meniu functionalitate 1
* p = 2(user input prin alegere\_optiune() )
* index = 3 prin citire\_index()
* input\_tuple = (13,14) print citire\_nr\_complex()
* apelare add\_cu\_index(input\_tuple[0], input\_tuple[1], index, numere\_complexe)
* se insereaza inaintea pozitei 4 (daca exista) numarul complex 13+14i

1. **Modifică elemente din listă:**

Funcții utilizate

|  |  |
| --- | --- |
| Denumire | Descriere |
| stergere\_numar(index, lista) | Sterge numarul de pe o pozitie data prin index in lista data; in cazul in care pe pozitia data nu exista un element se ridica o eroare si se afiseaza un mesaj |
| stergere\_interval(start, stop, lista) | Sterge numere dintr-un intervat dat prin start stop din lista data |
| inlocuire\_numar(inlocuit, inlocuitor, lista) | Se inlocuieste fiecare aparitie a numarului inlocui cu numarul inlocuitor in lista data si returneaza intr-o lista pozitia fiecarei aparitii inlocuitului |

Scenariu rulare pentru stergerea elementului de pe pozitia 2 din lista [(1+2j), (3+4j), (5+6j), (7+8j)]:

* Apelare main()
* p = 2 prin alegere\_optiune()
* p = 1 prin alegere\_optiune()
* index = 2 prin citire\_index()
* apelare stergere\_numar(index, numere\_complexe)
* numarul (5+6j) va fi sters de pe pozita 2
* lista finala va fi [(1+2j), (3+4j), (7+8j)]

Scenariu rulare pentru stergerea elementelor din intervalul [1,3) din lista [(1+2j), (3+4j), (5+6j), (7+8j)]:

* Apelare main()
* p = 2 prin alegere\_optiune()
* p = 2 prin alegere\_optiune()
* start = 1 prin citire\_index()
* stop = 3 prin citire\_index()
* apelare stergere\_interval(start, stop, numere\_complexe)
* numerele (3+4j), (5+6j) vor fi sterse de pe pozitiile 1,2
* lista finala va fi [(1+2j), (7+8j)]

Scenariu de rulare pentru inlocuirea numarului (1+2j) cu (3+4j) din lista [(1+2j), (5+7j), (1+2j), (24+13j), (1+2j)]:

* Apelare main()
* p = 2 prin alegere\_optiune()
* p = 2 prin alegere\_optiune()
* inlocuit = (1+2j) prin citire\_nr\_complex()
* inlocuitor = (3+4j) print citire\_nr\_complex()
* apelare inlocuire\_numar(inlocuit, inlocuitor, numere\_complexe)
* i = 0 lista[i]==(1+2j) = > lista[i] = (3+4j) , lista\_aux[0] = 0
* i = 2 lista[i]==(1+2j) = > lista[i] = (3+4j) , lista\_aux[1] = 2
* i = 4 lista[i]==(1+2j) = > lista[i] = (3+4j) , lista\_aux[2] = 4

**3.Căutare numere**

Funcții utilizate

|  |  |
| --- | --- |
| Denumire | Descriere |
| parte\_imag(start, stop, lista) | Returneaza o lista formata din partile imaginare din numerele din lista data din secventa data de start si stop |
| modul\_mai\_mic\_ca10(lista) | Returneaza o lista formata din toate numerele complexe cu modulul mai mic ca 10 din lista data |
| modul\_egal\_cu10(lista) | Returneaza o lista formata din toate numerele complexe cu modulul egal cu 10 din lista data |

Scenariu de rulare pentru afisarea parti imaginare a numerelor din secventa 0, 3 din lista [(1+2j), (3+4j), (7+8j)]:

* apelare main()
* p = 3 din alegere\_optiune()
* p = 1 din alegere\_optiune()
* start = 0 din citire\_index()
* stop = 3 din citire\_index()
* apelare parte\_imag(start, stop, numere\_complexe)
* i = 0, lista\_aux[0] = 2.0
* i = 1, lista\_aux[1] = 4.0
* i = 2, lista\_aux[2] = 8.0
* se afiseaza lista\_aux == [2.0, 4.0, 8.0]

Scenariu de rulare pentru afisarea numerelor complexe cu modulul mai mic ca 10 din lista [(1+2j), (3+4j), (6+8j)]:

* apelare main()
* p = 3 din alegere\_optiune()
* p = 2 din alegere\_optiune()
* apelare modul\_mai\_mic\_ca10(numere\_complexe)
* item = (1+2j) cu modul mai mic ca 10, lista\_aux[0] = (1+2j)
* item = (3+4j) cu modul mai mic ca 10, lista\_aux[1] = (3+4j)
* se afiseaza lista\_aux = [(1+2j), (3+4j)]

Scenariu de rulare pentru afisarea numerelor complexe cu modulul egal cu 10 din lista [(6+8j), (8+6j), (2+3j), (5+6j)]:

* apelare main()
* p = 3 din alegere\_optiune()
* p = 3 din alegere\_optiune()
* apelare modul\_egal\_cu10(numere\_complexe)
* item = (6+8j) cu modul egal cu 10, lista\_aux[0] = (6+8j)
* item = (8+6j) co modul egal cu 10, lista\_axu[1] = (8+6j)
* se afiseaza lista\_aux = [(6+8j), (8+6j)]