Noțiuni introductive

- informații → date → structuri de date (SD)
- sistemele de calcul ocupă mult timp cu:
  - stocarea datelor (S)
  - accesarea datelor (A)
  - manipularea datelor (M)
- reprez. obiectelor din lumea în aplicații software:

1) nevoia $\xrightarrow{\text{MODELARE}}$ entități matematice
2) mulțimea operațiilor pe entitățile de la 1
3) maniera în care entitățile de la 1) sunt reprezentate și stocate în memorie

Tipuri abstracte de date

- Tip de date
  - domeniu (structura / implementarea)
  - operații asociate
- Un tip abstract de date este un tip cu următoarele proprietăți:
  1. specificarea obiectelor este independentă de reprezentarea lor

, adică utilizatorii TAD nu trebuie să știu cum sunt efectiv implementate obiectele

2. specificarea operațiilor este independentă de implementarea lor
, adică utilizatorii TAD nu trebuie să știu cum sunt efectiv realizate aceste operații în interior

· Domeniul unui TAD poate fi definit prin enumerarea elementelor sale în cazul în care este finit, fie printr-o regulă care descrie elementele sale.
· După definirea domeniului unui TAD este necesară specificarea operațiilor (date, rezultate, precondiții, postcondiții)

Interfața unui TAD

Tipurile de operații din interfața unui TAD sunt:
· operații de creare a elementelor de acel tip
· operații de distrugere a elementelor de acel tip
· operații de accesare a componentelor instanțelor
· operații de manipulare a instanțelor
· operații specifice tipului de date

Folosirea abstractizării și încapsularea în proiectarea programelor

1. **Încapsularea datelor** (ascunderea informației) este definită ca fiind ascunderea detaliilor de implementare ale unui obiect.

2. **Abstractizarea** datelor este definită ca fiind separarea dintre specificarea unui obiect și implementarea lui

Folosirea abstractizării și încapsulării ajută la:

- dacă avem nevoie în faza de proiectare de tipurile de date A, B, C, ... vom avea nevoie doar de specificații
- fiecare din tipurile A, B, C, ... pot fi testate și verificate separat
- reutilizarea - extragerea unei structuri de date dintr-o aplicație și folosirea acesteia în altă aplicație
- se poate schimba reprezentarea unui tip de date fără a afecta alte aplicații care folosesc acel tip de date cu condiția ca operațiile tipului să nu fie modificate

Structuri de date

- Domeniul structurilor de date (SD) se ocupă cu stocarea și accesarea datelor
- O SD se poate considera din p.d.v.:
  1. Logic, ca și definiție (elementele ei constitutive și leg. între ele)
  2. fizic, ca mod de memorare (stocare)
     (a) Structură statică ocupă în memorie o zonă de dimensiune constantă, exemple: tablouri, articole
     (b) Structură semistatică ocupă în memorie o zonă de dimensiune constantă, dar elementele ocupă loc variabil în timpul execuției, exemple: tabele de dispersie
     (c) Structuri de date dinamice ocupă în memorie o zonă care se alocă dinamic în timpul execuției programului fără a avea o dimensiune constantă, exemple: liste înlănțuite, arbori

TAD/Container ⟵	SD
Vector Dinamic	tablou
Matrice	lista înlănțuită
Colecție	- simplu
Mulțime	- dublu
Dicționare	- alocare dinamică
Dicționar, Dicționar ordonat,	- înlănțuiri pe tablou
Multidicționar, Multidicționar ordonat	tabela de dispersie
Lista	- liste independente
Lista ordonată	- liste întrepătrunse
Coada	- adresare deschisă
Stiva	ansamblu
Coada cu priorități	arbore binar de căutare
Arbore (binar)	- arbore echilibrat (AVL)

SD - definesc cum sunt stocate și accesate datele din memorie fără a impune o implementare

TAD - implementări ale SD-urilor