Tema 2. Algoritmi: caracteristici, reprezentare, implementare

Obiective

- **să recunoşti tipurile de blocuri ale schemelor logice**
- **să corelezi blocurile din schemele logice cu instucțiunile din limbajul pseudocod.**

Fişa de documentare 2.2. Reprezentarea algoritmilor: Scheme logice. Limbaj pseudocod.

Reprezentarea algoritmilor

După etapa de analiză a problemei în care s-au stabilit datele de intrare şi cele de ieşire, urmează etapa de **elaborare a algoritmului**.

Acesta trebuie reprezentat într-un mod inteligibil. Întrebarea este cum putem să reprezentăm algoitmul astfel încât să fie înteles de cei ce îl citesc?

Un posibil răspuns ar fi – printr-un limbaj de programare. Este un raspuns bun pentru cei ce cunosc acel limbaj de programare, însa ceilalți nu vor întelege nimic. Nu putem să impunem altora să învete acel limbaj doar pentru a intelege algoritmii descrişi de noi. În plus, se observă că nu există niciun limbaj de progamare care să dureze sau care să fie acceptat de toata lumea.

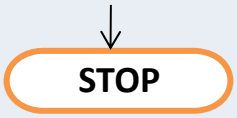
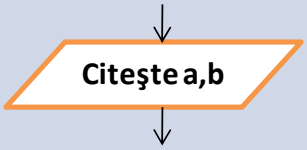
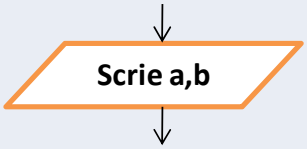
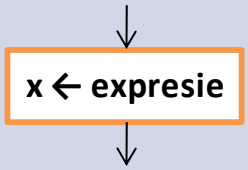
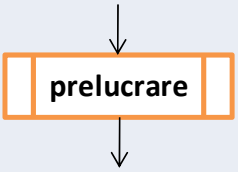
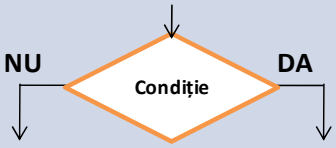
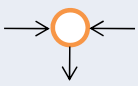
Este deci necesară utilizarea unui limbaj comun de repezentare a algoritmilor, dând apoi posibilitatea fiecarui programator să "traducă" algoritmul în ce limbaj de programare doreşte.

De-a lungul timpului s-au remarcat două modalități de reprezentare a algoritimilor: **schemele logice** şi **limbajul pseudocod**.

Schemele logice reprezintă un algoritm în mod grafic, folosind blocuri diferite pentru operații diferite. Această metodă are unele dezavantaje: schemele sunt stufoase, greu de urmărit. În tabelul urmator prezentăm tipurile de blocuri folosite în reprezentarea algoritmilor.

Schemele logice sunt mai utile celor care abia învață să programeze şi deci sunt în **faza de formare a gândirii algoritmice**. Recomandăm ca la scrierea schemelor logice să se scrie mai întâi conținutul blocului şi apoi să se deseneze blocul corespunzător.

Blocurile specifice schemelor logice

SIMBOL	DENUMIRE	SEMNIFICAȚIE
	**Bloc terminal**	Marchează începutul algoritmului
	**Bloc terminal**	Marchează sfârşitul algoritmului
	**Bloc de intrare / citire a datelor de intrare**	Se face transferul de date de la utilizator către algoritm
	**Bloc de ieşire / scriere a datelor de ieşire**	Se face transferul de date către utilizator
	**Bloc de atribuire**	Variabilei x i se atribuie valoarea expresiei
	**Bloc de prelucrare** O prelucrare este compusă din mai multe instrucțiuni elementare	În interior se scriu instrucțiunile sau un nume ce desemnează un grup de instrucțiuni
	**Bloc de decizie** în care se evaluează **condiția** obținându-se o valoare logică “Adevarat” sau “Fals”	Dacă condiția este adevărată se execută ramura “**DA**”, altfel se execută ramura “**NU**”
	**Bloc conector logic**	Conectează mai multe puncte din algoritm
	**Bloc conector de pagină**	Specifică pagina cu care se continuă schema

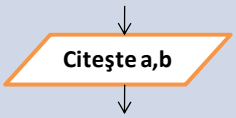
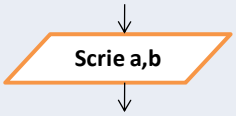
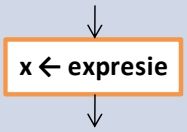
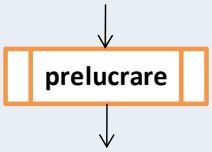
Limbajul Pseudocod

Limbajul pseudocod este un ansamblu de convenții (codificări) care definesc operațiile (instrucțiunile) permise pentru reprezentarea algoritmilor. Respectand aceste convenții, chiar în forme diferite, algoritmii reprezentați în pseudocod pot fi citiți de orice persoană, indiferent că este sau nu programator.

Limbajul pseudocod nu respectă o sintaxa anume, nu are un standard. Sunt doar nişte convenții pe care trebuie să le respectăm atunci când reprezentăm un algoritm. Înstructiunile se pot scrie în limba engleză sau în limba română. În acest material şi în cele ce urmează vom adopta un limbaj cu instrucțiuni în limba română.

În pseudocod putem scrie declarări de variabile, specificând numele şi tipul lor. Pe lângă declarațiile de variabile, limbajul pseudocod conține cuvinte cheie, instrucțiuni (început, sfârşit, intrare/ieşire, atribuire, decizie, selecție, repetitive), proceduri/funcții. Toate acestea vor fi prezentate pe larg în fişa de documentare următoare.

Prezentăm mai jos corespondența între instrucțiunile pseudocod şi blocurile din schemele logice.

INSTRUCȚIUNE PSEUDOCOD	DENUMIRE	SIMBOL SCHEMĂ LOGICĂ
a, b, c intregi x,z reale	**Declaratii de variabile**	–
Citeşte a,b	**Citirea datelor de intrare**	
Scrie a,b	**Scrierea datelor de ieşire**	
x ← 10 a ← a+1	**Instrucțiune de atribuire**	
| instructiune1 | instructiune2 | instructiune3 |_■	**Bloc de Instrucțiuni** O prelucrare este compusă din mai multe instrucțiuni elementare	
┌**dacă** c **atunci** | execută p |**altfel** | execută q |■	**Instrucțiune de decizie** în care se evaluează **condiția** obținându-se o valoare logică “Adevarat” sau “Fals”