

## SISTEME DE NUMERAȚIE

**Determinarea** se face prin împărțirea succesivă a valorii exprimate în sistemul de numerație zecimal –  $N_{10}$  – la noua bază  $q$  (în cazul dat:  $q=2$  respectiv  $q=16$ ); deîmpărțitul este mai întâi numărul în sine, apoi succesiv câtul rezultat). Se colectează resturile obținute și (atenție, a nu se uita) ultimul cât rămas (evident  $< q$ , acesta fiind coeficientul  $a_n$ ). În cazul în care noua bază  $q > 10$ , se convertesc numerele din acest șir într-o singură cifră din noua bază (la hexazecimal  $q=16$ ; exemplu: la valoarea restului de 13, coeficientul  $a_j$  oarecare, va fi D).

**Reprezentarea** numărului în această bază  $q$  se obține prin colectarea și alăturarea – în ordine inversă obținerii – a acestor cifre convertite:  $N_2 = a_n a_{n-1} a_{n-2} \dots a_1 a_0$ .

**Verificarea exactă** se poate face prin dezvoltare polinomială:

$$N_2 = a_n \cdot q^n + a_{n-1} \cdot q^{n-1} + a_{n-2} \cdot q^{n-2} + \dots + a_1 \cdot q^1 + a_0 \cdot q^0 = N_{10} !!$$

Orientativ se poate face o verificare preliminară a ordinului de mărime. Numărul dat, trebuie să îndeplinească condiția:  $N_{10} < q^m$ , unde  $m$  este numărul total al coeficienților  $a_j$ :  $m = n + 1$ .

## ALGEBRĂ BOOLEANĂ (LOGICĂ). OPERAȚII LOGICE FUNDAMENTALE

### **Definire/Enunț**


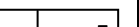
**Negația** logică schimbă valoarea de adevăr a unei propoziții/variabile logice: produce o valoare de adevăr fals dacă (și numai dacă!) propoziția în cauză este adevărată și invers, rezultatul este adevărat dacă (și numai dacă!) propoziția este falsă.

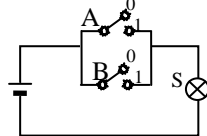
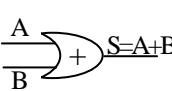
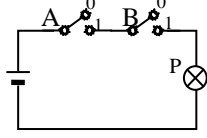
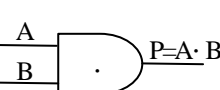
**Suma** logică este acea operație care aplicată la una sau mai multe propoziții adevărate produce un rezultat adevărat; rezultatul este fals dacă și numai dacă toate propozițiile sunt false.

**Produsul** logic are valoarea adevărat numai dacă toate propozițiile/argumentele sunt adevărate și fals dacă cel puțin una/unul este falsă/fals.

**Reprezentare prin tabele de adevăr, o cuprinde pe cea simbolică și modelarea.**

S-au utilizat simbolurile: 0 pentru Fals și 1 pentru Adevărat

Negația	Cuvânt-cheie: NU (NOT)		Circuit-model	Simbol grafic
	Simbolizare: $\bar{A}$			
A	0	1		
$\bar{A}$	1	0		
$A \Rightarrow$	1	0		

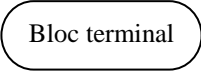


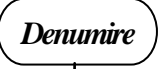


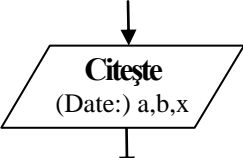
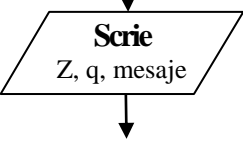
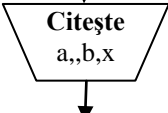
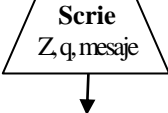
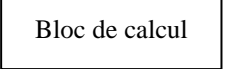
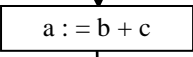
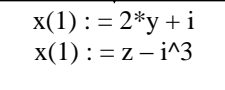
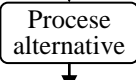
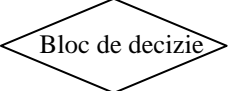
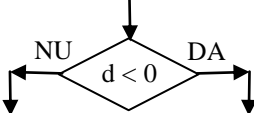
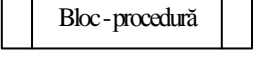
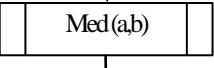
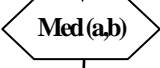


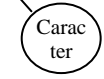




Suma	Cuvânt-cheie: SAU (OR)				Circuit-model	Simbol grafic
	Simbolizare: S=A+B; S=A∨B					
A	0	0	1	1		
B	0	1	0	1		
S ⇒	0	1	1	1		
Produsul I	Cuvânt-cheie: ȘI (AND)				Circuit-model	Simbol grafic
	Simbolizare: P=A·B; P=A∧B					
A	0	0	1	1		
B	0	1	0	1		
P ⇒	0	0	0	1		

## ALGORITMI ȘI REPREZENTAREA/DESCRIEREA LOR

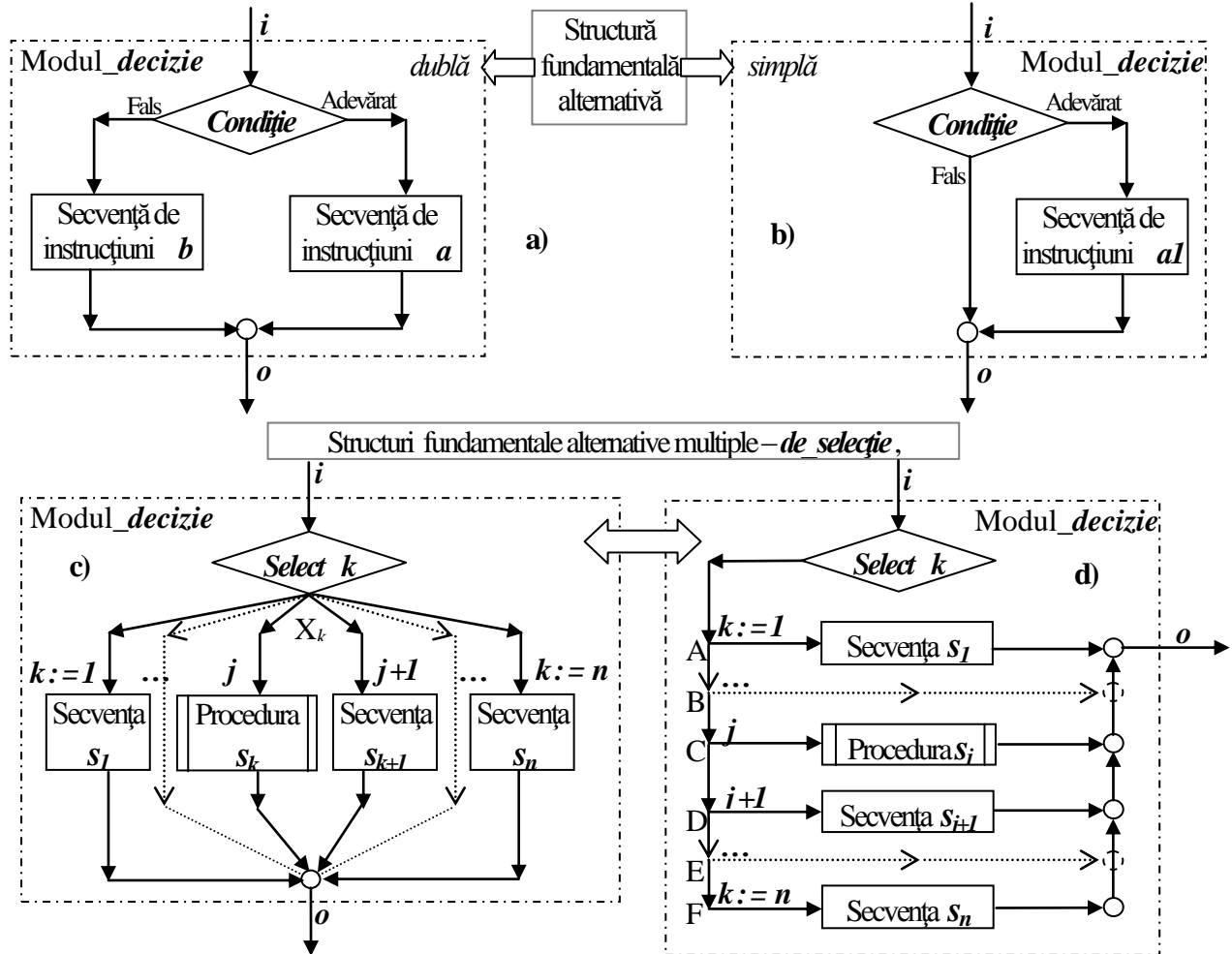
### **Limbaajul simbolic (schemele logice)**

Prin schemă logică se denumește o reprezentare de tip grafic a algoritmilor cu ajutorul unor figuri geometrice (denumite și simboluri) a căror formă indică tipul acțiunii care se execută în etapa pe care o simbolizează. Figurile poartă denumirea de blocuri. Blocurile sunt legate unul după altul, în ordinea execuției lor, prin segmente orientate. În interiorul lor se înscriu operațiile concrete din cadrul acțiunii care urmează a fi executată.

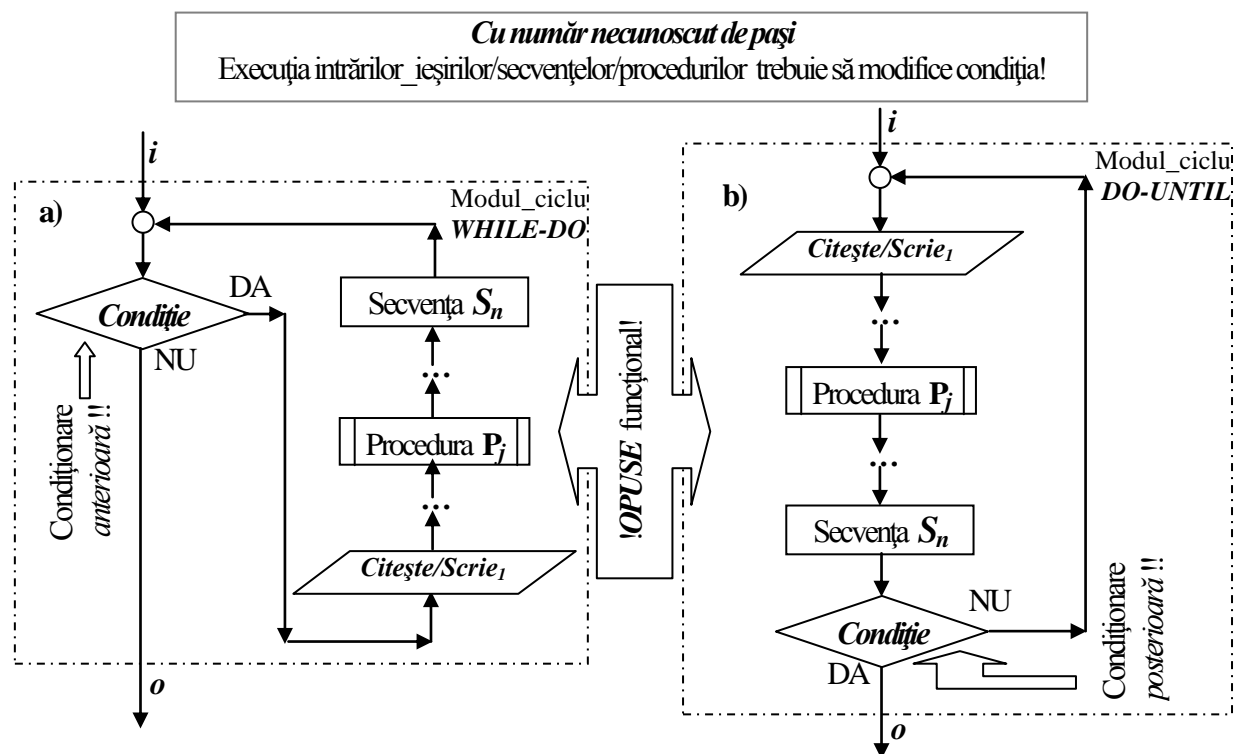
## Tipuri uzuale de blocuri

Nr. crt.	Simbol și denumire	Exemple	Rol în schema logică	Observații	Alternative
1	2	3	4	5	6
1		 	<p><u>Început</u> de schemă logică</p> <p><u>Sfârșit</u> de schemă logică</p>	<p>O singură ieșire! La subprograme apare și denumirea acestora</p> <p>O singură intrare! La subprograme apare <u>comanda de întoarcere</u>: RETURN!</p>	 
2		 	<p>Specifică <u>date</u> de intrare (inițiale)</p> <p>Specifică <u>date</u> de ieșire (rezultate), mesaje etc.</p>	<p>O intrare și o ieșire. Se diferențiază prin cuvântul cheie „boldat” <b>Citește</b></p> <p>O intrare și o ieșire. Se diferențiază prin cuvântul cheie „boldat” <b>Scrie</b></p>	 
Pt. suporturi externe particulare: Tabelul nr.III 7					
3		 	Specifică uzual una su mai multe <u>relații de calcul</u>	<p>O intrare și o ieșire</p> <p>O intrare și o ieșire</p>	
4			<p>Specifică o condiție</p> <p>Răpsunsul <u>logic</u> <i>Adevărat/Fals</i> (la întrebarea „se îndeplinește condiția?”) determină ramificația/traseul de urmat!</p>	<p>O intrare și <u>mai multe</u> ieșiri!</p>	
5			Specifică denumirea procedurii (=subprogram)	De regulă se specifică și argumentele (pt. o) intrare-(pt. o) ieșire	
6			Reunifică mai multe trasee ramificate	Mai multe intrări și o singură ieșire!	
7*	<p>Conector de ramificație la distanță</p> 	<p>Plecare spre</p>  <p>Sosire de la</p> 	Înlănțuiește (leagă) blocuri distanțate dintr-o <b>ramificație!</b>	Realizează un traseu de <u>ramificație</u> lung, ( <u>nedesenat</u> ) nominalizat cu caracterul $\alpha$	Se dispun câte două (în tandem): un bloc cu o intrare, în corespondență cu un bloc cu o ieșire
8			Evidențiază sensul traseelor de execuție a schemei logice (și succesiunea de parcurgere a blocurilor)		

**Exemple de structuri fundamentale alternative:**



**Exemple de structuri fundamentale repetitive:**



**Exercițiu:** să se transforme structura WHILE-DO din stânga într-una cu număr cunoscut de pași!

## PROCEDURI ÎN VISUAL BASIC

Procedurile în Visual Basic pot fi de tip subrutină sau funcție.

Aplicațiile Windows sunt constituite din segmente de cod, subrutine, ce răspund la anumite acțiuni specifice efectuate asupra unui control. Numele implicit al unei subrutine are sintaxa:

**NumeControl\_NumeEveniment**

Subrutinele, în general, nu returnează un anumit rezultat. Funcțiile execută o serie de instrucțiuni și returnează un anumit rezultat. Atât subrutinele, cât și funcțiile acceptă argumente, ce se transmit la procedură în momentul apelării acesteia.

Datorită returnării unui rezultat, funcțiile sunt de un anumit tip, în conformitate cu acesta (ex. Integer).

Transferul argumentelor se poate face prin referință (ByRef) sau prin valoare (ByVal). La transferul prin valoare, procedura vede doar o copie a argumentului. Eventualele modificări ale valorilor argumentelor nu se returnează aplicației. Transferul prin referință permite modificarea valorii argumentului respectiv în aplicație. În mod implicit, o subrutină are două argumente, transferate prin valoare:

- sender – obiectul ce reprezintă controlul care a declanșat subrutina;
- e – oferă informații suplimentare despre eveniment.

## TIPURI DE VARIABLE ÎN VISUAL BASIC

O variabilă are un nume și valoarea/valorile asociate. O clasificare a variabilelor se poate face în funcție de valoarea stocată:

- variabile numerice;
- variabile alfanumerice sau de tip șir;
- variabile logice sau de tip boolean;
- variabile tip dată (stochează data și ora);
- variabile tip obiect.

În funcție de accesibilitatea sau vizibilitatea variabilelor există variabile locale și globale. O variabilă declarată în interiorul unei proceduri, se poate accesa doar din procedura respectivă, fiind o variabilă locală. Variabilele ce pot fi accesate de mai multe proceduri trebuie declarate în afara procedurilor, în secțiunea de declarații a modulului sau clasei. Declararea variabilelor locale se face cu cuvântul cheie **Dim** sau **Private**. Variabilele globale se pot accesa din orice clasă și se declară cu cuvântul cheie **Public**.

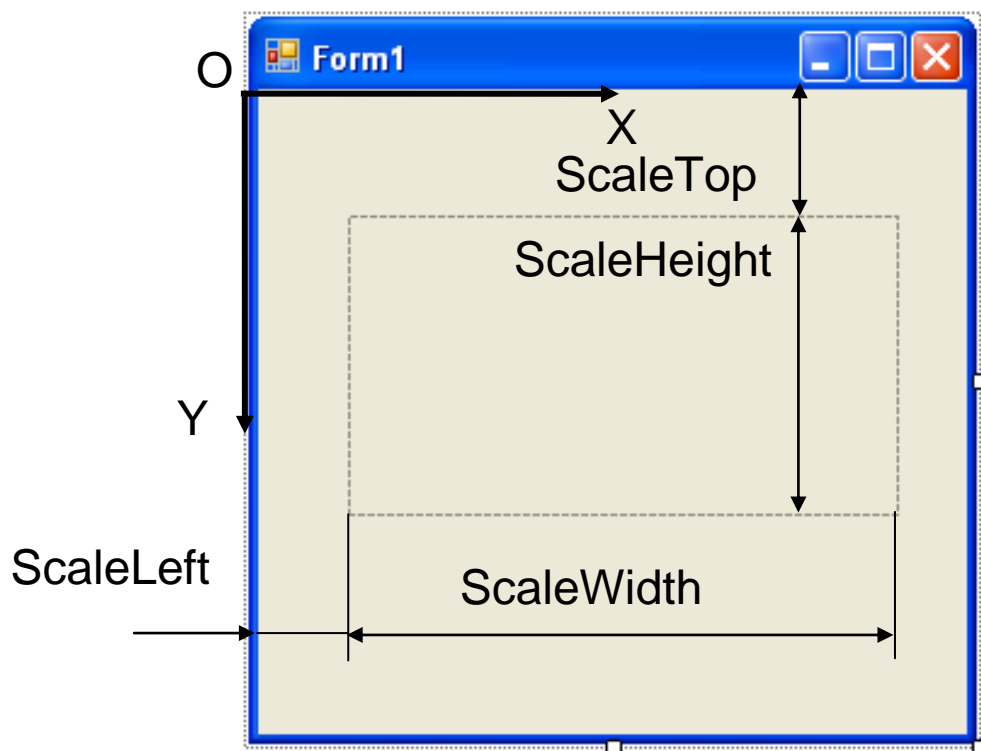
## CONCEPTUL “OBJECT ORIENTED” ÎN PROGRAMAREA VISUAL BASIC

Conceptul “object oriented” provine din faptul că aplicațiile programate în VB conțin instanțe ale unor clase numite “controale” care au proprietăți moștenite și care pot fi asociate cu secvențe de cod care se execută la apariția unui anumit “eveniment”. Evenimentele pot fi asociate cu periferia calculatorului (mouse, tastatură) sau cu diverse momente ale derulării aplicației (GetFocus, LostFocus)

## OBIECTELE CU CARACTER GRAFIC FOLOSITE ÎN PROGRAMAREA VISUAL BASIC

Obiectele cu caracter grafic în programarea VB sunt formurile, chenarele grafice și imprimantele (form, picturebox, printer)

## Dimensiunile obiectelor grafice și variabilele asociate lor în programarea VB



### FUNCȚIA SELECT FOLOSITĂ ÎN PROGRAMAREA BAZELOR DE DATE SQL

SELECT lista\_sectie (coloanele care se selecteaza)  
 FROM lista\_tabele (lista tabelor care contin coloanele)  
 WHERE prima\_restrictie (primul set de conditii ale randurilor selectate)  
 GROUP BY grupare\_coloane (modul de grupare a rezultatelor)  
 ORDER BY sortare\_coloane (ordinea de afisare a coloanelor)  
 HAVING restrictie\_secundara (conditii suplimentare pentru randurile selectate)  
 LIMIT numar1, numar2 (afiseaza 'numar2' linii dupa primele 'numar1');

Exemplu:

SELECT nume, adresa FROM elev WHERE nume LIKE "M%" ORDER BY nume LIMIT 10,5;

### CONECTAREA LA BAZELE DE DATE SQL CU AJUTORUL FUNCȚIILOR PHP

• **Conectarea** la serverul MySQL:

**mysql\_connect(nume\_gazda, nume\_utilizator, parola)**

Exemplu: \$db=mysql\_connect("localhost", "user", "parola") or die ("Nu s-a reușit conectarea");  
 echo "Conectarea la serverul de baze de date a fost reusita";

• **Selectia** unei baze de date:

**mysql\_select\_db(baza\_de\_date)**

Exemplu: \$ok=mysql\_select\_db("nume\_BD"); if (!\$ok) die ("Nu s-a reusit conectarea la baza de date"); else echo "Conectarea la baza de date a fost reusita";

• **Inchiderea conexiunii** cu serverul de baze de date:

**mysql\_close(nume\_conexiune)**

Exemplu: mysql\_close(\$db);

### UTILIZAREA FORMURILOR HTML ȘI PRINCIPALELE CONTROALE FOLOSITE ÎN CADRUL LOR

• **Structura unui formular HTML** este descrisa intre etichetele <form> si </form>;

•Datele din formular se prelucreaza in urma executiei comenzii METHOD care are structura:

<form METHOD="GET|POST" ACTION="URL" >..... </form>;

•Incorporarea controalelor se face cu comanda INPUT:

<INPUT TYPE="tip\_de\_control" NAME="nume\_de\_control" [OPTIUNI]

VALUE="valoare">;

INPUT TYPE="TEXT" poate stoca o singura linie de text

INPUT TYPE="TEXTAREA" poate stoca mai multe linii de text (oricate)

INPUT TYPE="PASSWORD" poate stoca un sir mascat de caractere cu valoare de password;

## PROBLEMA

Să se explice construirea unei aplicații (specificare controale, evenimente) în Visual Basic care să indice ora locală și ora GMT.

Interfața grafică trebuie să conțină un control pentru afișarea rezultatului, care poate fi o casetă text sau o etichetă. Mai sunt necesare 2 butoane, unul pentru afișarea orei locale și unul pentru afișarea orei GMT. Se mai poate introduce un buton pentru închiderea aplicației. În subrutina de acționare a butoanelor (eveniment click) se introduce codul pentru setarea proprietății text a controlului de afișare.