

Soluția cea mai sigură este să facem matricea de adiacență pe care să o ridicăm la puterea a 2a ca să obținem TOATE drumurile de lungime 2:

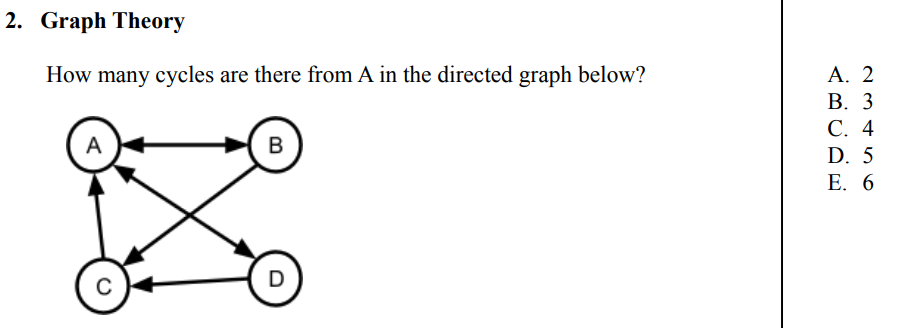
Ne amintim că se ia fiecare linie din prima matrice cu fiecare coloană din a doua.

Elementele de pe aceleași poziții se înmulțesc și rezultatele se adună.

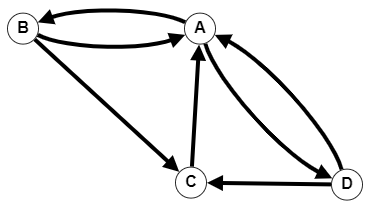
Pentru exemplificare, am colorat cu roșu linia 1 și coloana 1, rezultatul lor se pune în C[1,1] = 0\*0+1\*1+0\*1+0\*0+1\*0

DECI numărul total de drumuri se obține însumând TOATE valorile din matricea asta:

15 Răspuns: D 15



Ca idee - se văd mai bine ciclurile dacă înlocuiți fiecare săgeată cu 2 capete cu două care sunt fiecare în sensuri opuse:



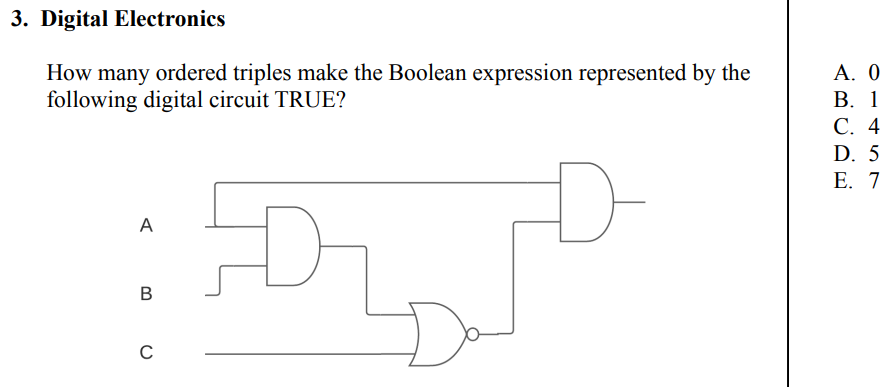
Ciclurile sunt (ne reamintim că n-avem voie să repetăm muchii și, la ACSL, nici noduri (în afară de cel care închide ciclul) și că dacă scriem același ciclu plecând din alt nod NU se consideră că e diferit (e același de fapt)

ABA

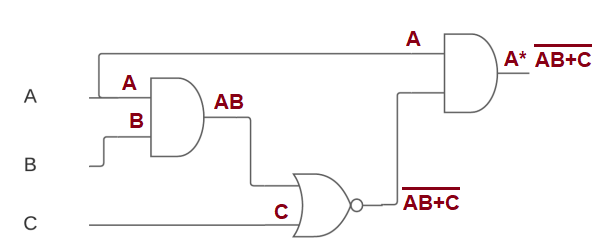
ABCA

ADA

ADCA Răspuns: C 4



Ne rescriem circuitul punând pe sârme valorile respective:



DECI valoarea finală a expresiei care traduce circuitul este

A\*()

DECI avem un produs care tre' să fie egal cu 1.

Acest lucru se-ntâmplă doar când ambii membri sunt 1.

Aflăm deci că A=1

Al doilea membru devine

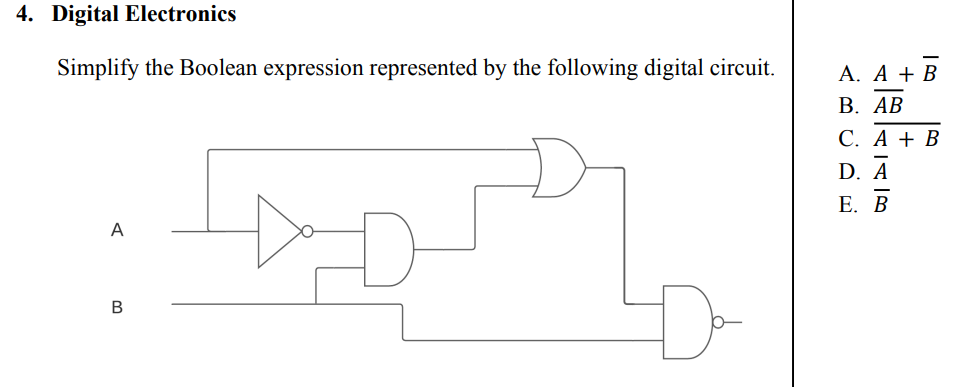
deci B+C=0 lucru care se-ntâmplă doar când B=0 C=0.

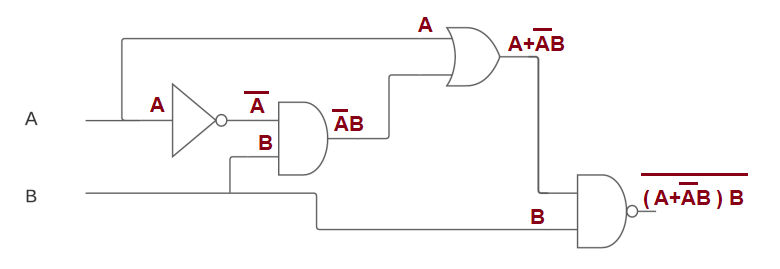
DECI e o singură soluție: A=1, B=0, C=0 RĂSPUNS: B. 1

Obs: Soluția noastră s-a bazat pe niște cazuri ceva mai particulare.

Rezolvarea generală funcționează întotdeauna dacă ne scriem tabelul de valori și calculăm expresia pentru fiecare triplet posibil:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| A  (1) | B  (2) | C  (3) | AB  (4)=(1)\*(2) | AB+C  (5)=(4)+3 | (6)= | (7)=(1)\*(6) |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 |





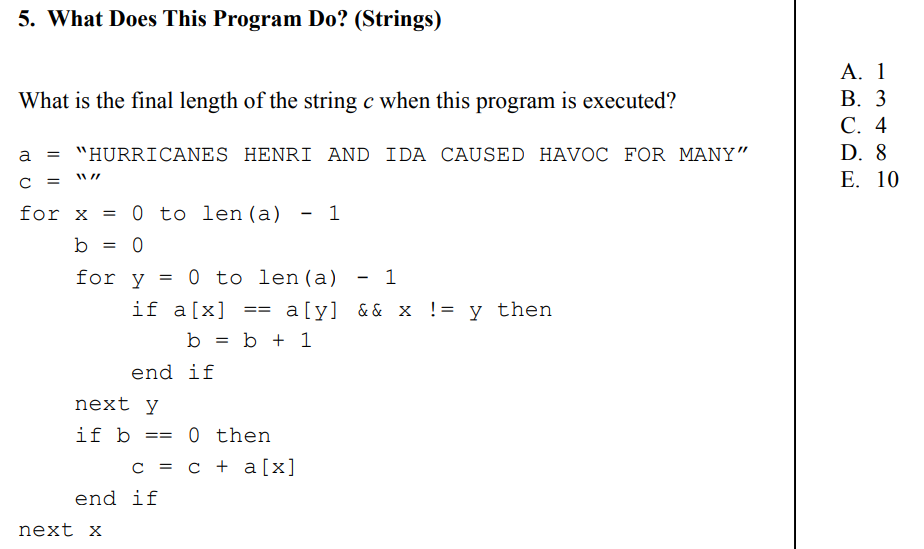
DECI trebuie să simplificăm expresia finală

Mai întâi luăm:

(ne folosim de distributivitatea lui + față de \* care există în algebra booleană, dar care Nu există în aritmetica zecimală: A + BC = (A+B)(A+C) )

Expresia finală este deci

DECI răspuns corect: E



Avem for pe for, cu variabilele x și y, are ambele se plimbă printre toți indicii posibili din șirul de caractere.

Deci ia orice caracter din șir cu orice alt caracter.

DOAR dacă cele de pe poziții diferite (x!=y) sunt egale (a[x]==a[y]) le numără (se adună la b valoarea 1).

După fiecare caracter a[x] se verifică dacă b==0, adică dacă acesta NU se repetă.

Morala: la final șirul c conține DOAR caracterele care NU se repetă (care apar fix o singură dată în string)

Deci pe exemplul dat șirul c va avea conținutul: "VFMY"

Răspuns: C) 4