* Descrierea programului:

Abordare Divide et Impera pentru rezolvarea problemei 2, var. 3;

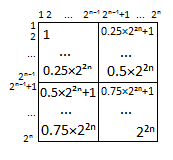
Complexitate – *O(kn)*, unde *k* este numărul de valori de aflat și *2ⁿ×2ⁿ* este dimensiunea matricii.

* Descrierea funcției recursive:

Considerăm parametrii funcției numărul *n* și valorile *x* și *y* ce reprezintă coloana, respectiv rândul pe care se află valoarea de calculat (în program s-a considerat și valoarea 2ⁿ ca parametru pentru a reduce complexitatea folosirii repetate a funcției *Math.pow(double,double)*);

* Cazul de continuare:

*n > 1:*



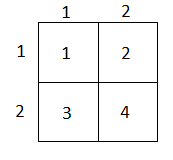
Când se reapelează funcția recursivă, matricea curentă degenerează într-unul din cadranele acesteia, iar coordonatele *x* și *y* se vor modifica corespunzător.

Se disting următoarele subcazuri:

1. (x,y) se află în cadranul stânga-sus: se reapelează funcția cu *n - 1, x* și *y* ca argumente;
2. (x,y) se află în cadranul dreapta-sus: valoarea returnată va fi mai mare decât *0.25×;* se reapelează funcția cu *n - 1, x –*  și *y* ca argumente și se adună *0.25×* la valoarea returnată;
3. (x,y) se află în cadranul stânga-jos: valoarea returnată va fi mai mare decât *0.5×;* se reapelează funcția cu *n - 1, x* și *y –*  ca argumente și se adună *0.5×* la valoarea returnată;
4. (x,y) se află în cadranul dreapta-jos: valoarea returnată va fi mai mare decât *0.75×;* se reapelează funcția cu *n - 1, x –*  și *y –*  ca argumente și se adună *0.75×* la valoarea returnată;

* Cazul de oprire:

*n = 1:*



În acest caz, *x* și *y* iau valori între 1 și 2; se returnează valoarea corespunzătoare acestora.

* Demonstrația complexității:

Folosind Teorema Master: *T(n) = aT(n/b) + f(n)*

În particular, pentru acest algoritm:

*T(m) = aT(m/b) + f(m),* unde *m =*

*a = 1* : se apelează funcția *select(int,double,double,double)* o singură dată în ea însăși;

*b = 2 :* dimensiunile matricii se împart la 2 la fiecare reapelare a funcției *select(int,double,double,double);*

*f(m) = Θ(1)* : restul funcțiilor din *select(int,double,double,double)* au complexitate constantă.

Rezultă din Teorema Master *T(m) = O() = O() = O(n)*.

Complexitatea *O(kn)* survine din faptul că aplicăm același algoritm pentru fiecare dintre cele *k* valori de calculat.