# Colegiul National “Liviu Rebreanu”

Lucrare de Atestat

Integrale definite

Absolventa: Profesor coordonator:

Badiu Anca Mureșan Vasile Ciprian

Cuprins

1. Introducere

2. Manual de utilizare

3. Modul de calcul

4. Codul sursă

5. Bibliografie

Introducere

Aplicatia **Integrale** este destinata elevilor de clasa a XII-a care doresc sa vizualizeze mai usor calcularea integralelor.

Sunt premise functii care contin adunari, scaderi, inmultiri, impartiri, ridicari la putere, radicali si urmatoarele functii trigonometrice:

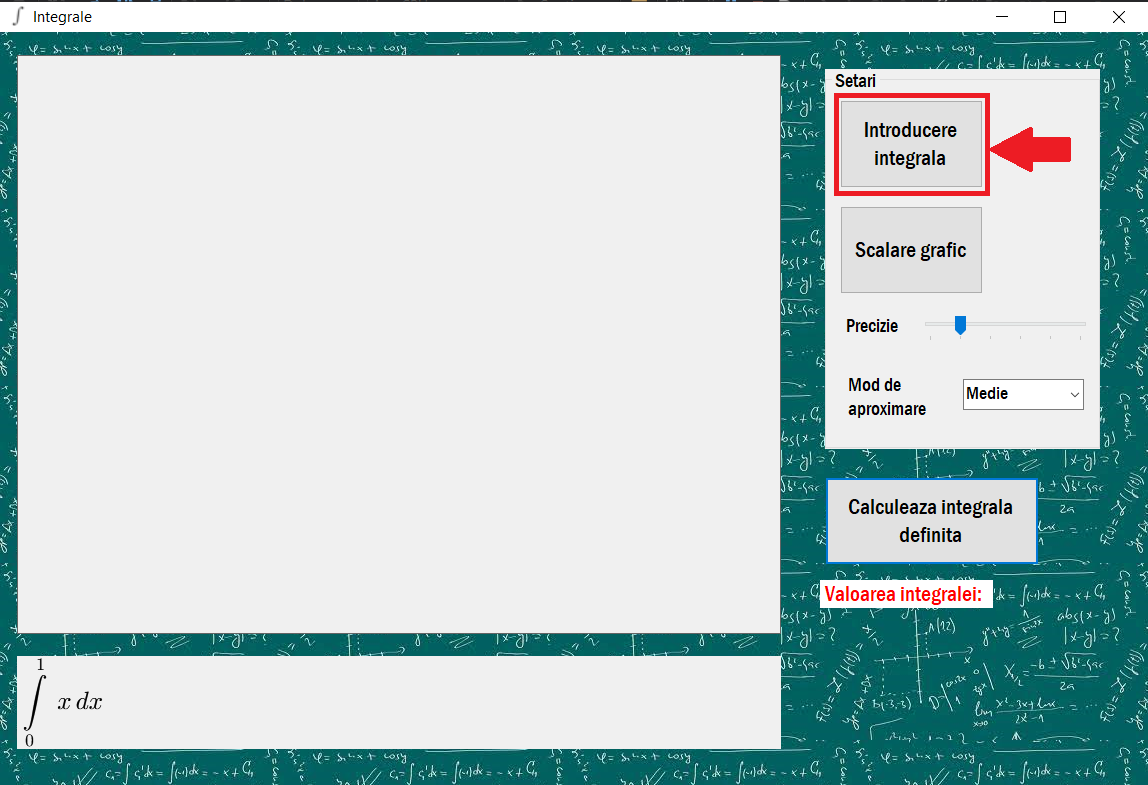
* sin(x), cos(x), tg(x), ctg(x)
* arcsin(x), arccos(x), arctg(x), arcctg(x)

Functiile pot contine si constanta pi si numarul lui Euler.

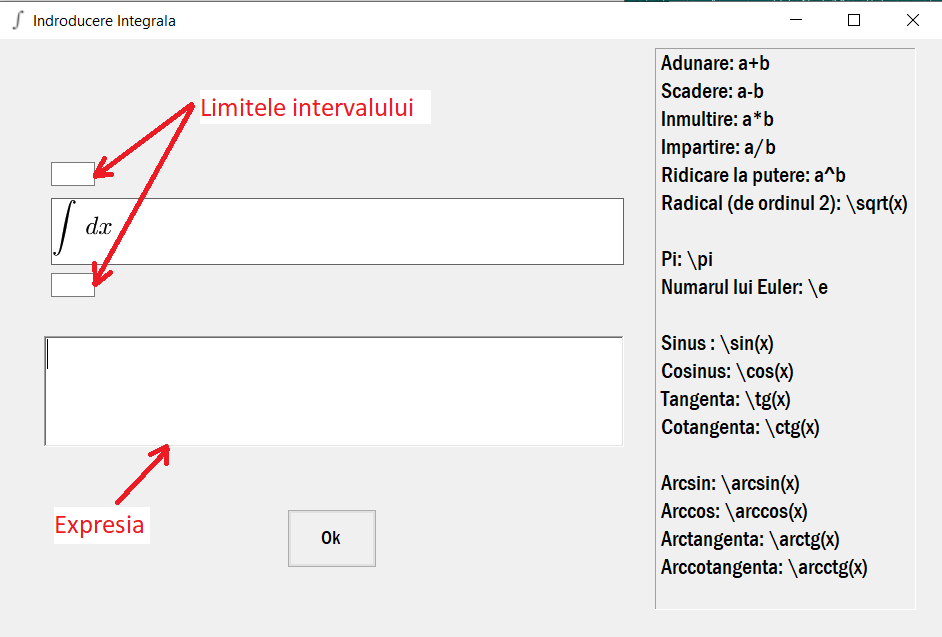
Aplicatia deseneaza functia introdusa si calculeaza integrala definita pe un interval dat.

Manual de utilizare

Pentru a introduce o integrala se va apasa butonul “Introducere Integrala” din dreapta sus din chenarul “Setari”.



Acesta va deschide o noua fereastra unde se pot introduce integrala si limitele intervalului pe care aceasta va fi calculata. Regulile de scriere vor aparea in dreapta, iar daca integrala este scrisa corect, aceasta va aparea intre casutele in care au fost introduse limitele intervalului.



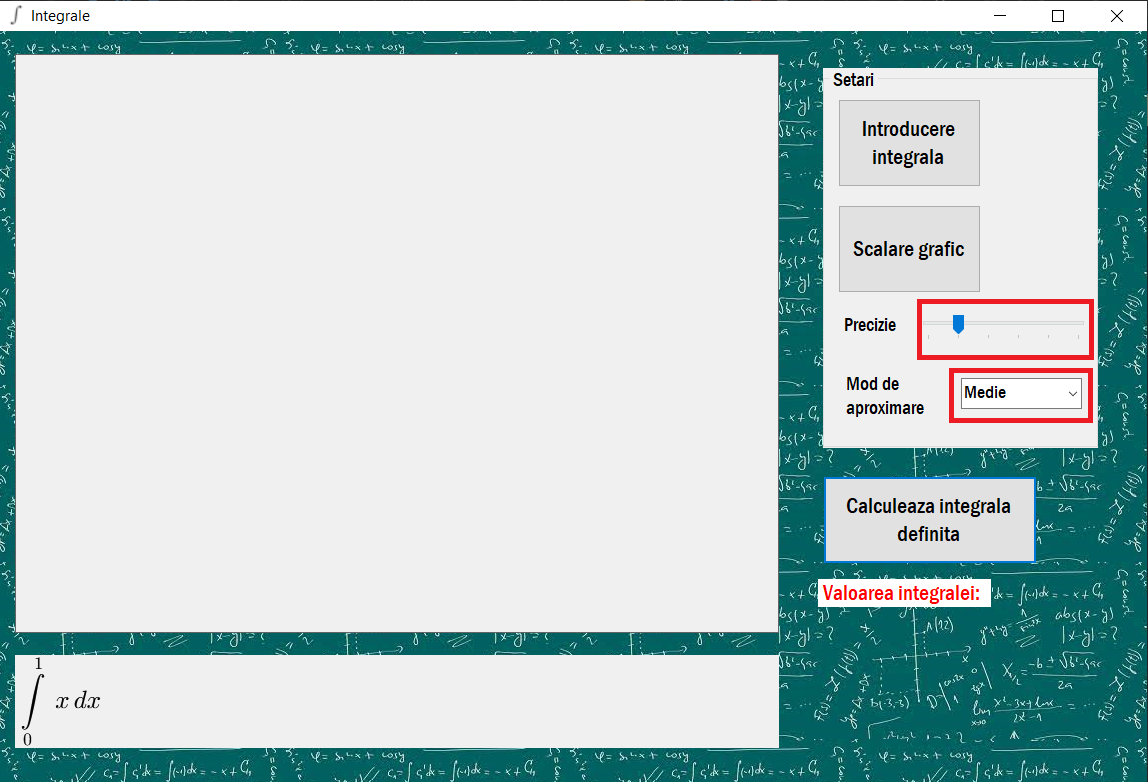
Pentru a confirma integrala introdusa, apasati “Ok”.

Acum integrala introdusa apare in josul formei. Se poate seta precizia cu care se calculeaza integrala (de la 1 la 6 zecimale) prin folosirea trackbar-ului. Se poate alege si modul de aproximare, care poate fi:

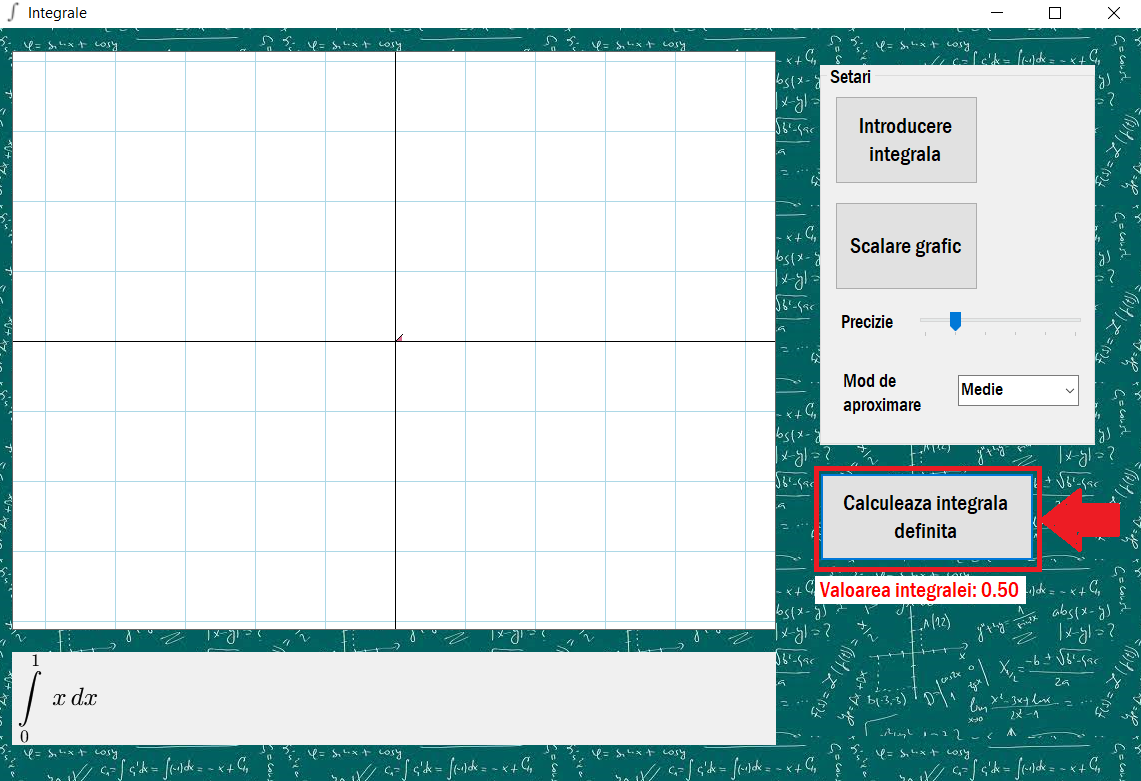
“Normal”: Calcularea integralei pe un subinterval se face prin calcularea functiei in ambele limite ale intervalului si calcularea mediei celor doua valori.

“Prima valoare”: Calcularea integralei pe un subinterval se face prin calcularea functiei in limita inferioara a intervalului.

“Ultima valoare”: Calcularea integralei pe un subinterval se face prin calcularea functiei in limita superioara a intervalului.

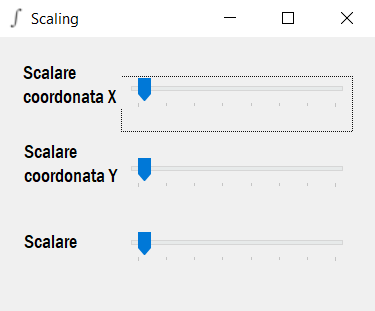


Pentru a calcula integrala, se apasa “Calculeaza integrala definita”. Graficul integralei va aparea in stanga formei, iar valoarea ei sub butonul apasat.



Se poate misca graficul prin tragerea acestuia.

Apasarea butonului “Scalare grafic” din chenarul “Setari” va deschide o forma unde putem controla dimensiunile functiei, in general, pe axa X sau pe axa Y.



Modul de calcul

Pentru a calcula integrala, impartim intervalul dat in intervale mai mici de lungimi egale cu precizia aleasa. Pentru fiecare subinterval, calculam functia in limita superioara, inferioara sau in ambele, depinde de modul de aproximare ales. Aceasta valoare o vom inmulti cu precizia, pentru a obtine integrala de pe subintervalul respectiv. Adunand aceste valori pentru toate subintervalele, vom obtine o aproximare pentru integrala data initial.

Pentru a calcula functia intr-un anumit punct trebie sa interpretam expresia data. Declaram matricea resultExpression[][] care are rolul de a retine rezulatul expresiei intre doi indici (resultExpression[i][j] e rezultatul expresiei de la i la j).

Initial, cunoastem unele valori ale acestei expresii cum ar fi constantele (e, π sau numere) sau variabila x in care calculam functia, astfel le vom introduce in matrice. E nevoie sa aflam resultExpression[0][n - 1] unde n este lungimea expresiei.

Pentru a tine cont de ordinea operatiilor folosim recursia: EvalExpression(int beg, int fin) care introduce in matrice valoarea expresiei dintre indicii beg si fin. Cand calculam aceasta expresie, calculam prima data rezultatul tuturor expresiilor din paranteze, apoi rezultatul functiilor (de exemplu sin sau radical), ridicarilor la puteri, inmultirilor si impartirilor, adunarilor si scaderilor, in aceasta ordine. Dupa efectuarea acestor calcule rezulatul expresiei dintre beg si fin va fi introdus in matrice.

Astfel EvalExpression(0, n - 1) va introduce in matrice rezultatul functiei in punctul x.

Cod sursa

* **Structura codului**

Pentru a afisa notatii matematice, folosim libraria WpfMath.

Codul contine 3 clase:

Expression: retine functii si le poate evalua in anumite puncte

Graph: deseneaza graficul

WpfExpression: interactioneaza cu WpfMath si exporta fisiere png corespunzatoare expresiilor date

* **Functia EvaluateAtP din clasa Expression:**

**public** **double** EvaluateAtP(**double** point, **out** **string** error)

{

p = point;

//resetting calculated parts

calcbeg = **new** **int**[expLength]; Array.Copy(nrbeg, calcbeg, expLength);

calcend = **new** **int**[expLength]; Array.Copy(nrend, calcend, expLength);

//put value instead of x

**for** (**int** j = 0; j < expLength; ++j)

**if** (expression[j] == 'x')

{

resultExpression[j][j] = p;

calcbeg[j] = calcend[j] = j;

}

**if** (EvalExpression(0, expLength - 1, **out** error)) **return** resultExpression[0][expLength - 1];

**else** **return** -1;

}

* **Functia EvalExpression din clasa Expression:**

**bool** EvalExpression(**int** beg, **int** fin, **out** **string** error)

{

//paranthesis

error = "";

**for** (**int** i = beg; i <= fin; i++)

{

**if** (expression[i] == '(')

{

**if** (!EvalExpression(i + 1, next[i] - 1, **out** error)) **return** **false**;

resultExpression[i][next[i]] = resultExpression[i + 1][next[i] - 1];

calcbeg[next[i]] = i;

calcend[i] = next[i];

i = next[i];

}

}

//functions

**for** (**int** i = beg; i <= fin; ++ i)

{

**try**

{

**if** (expression[i] == '\\')

{

**if** (!CalcFunction(**ref** i, **out** error)) **return** **false**;

}

}

**catch** (Exception e) { error = e.Message + " Expresie invalida::functii"; **return** **false**; }

}

//exponents

**for** (**int** i = beg; i <= fin; ++i)

{

**try**

{

**if** (expression[i] == '^')

{

**if** (!CalcOperation(**ref** i, **out** error)) **return** **false**;

}

}

**catch** (Exception e) { error = e.Message + " Expresie invalida::exponenti"; **return** **false**; }

}

//multiplication and division

**for** (**int** i = beg; i <= fin; ++i)

{

**try**

{

**if** (expression[i] == '\*' || expression[i] == '/')

{

**if** (!CalcOperation(**ref** i, **out** error)) **return** **false**;

}

}

**catch** (Exception e) {error = e.Message + " Expresie invalida::inmultire & impartire"; **return** **false**; }

}

//addition and subtraction

**for** (**int** i = beg; i <= fin; ++i)

{

**try**

{

**if** (expression[i] == '+' || expression[i] == '-')

{

**if** (expression[i] == '-' && (i == 0 || expression[i - 1] == '('))

{

**if** (calcend[i + 1] == -1)

{

error = "Expresie invalida::adunare & scadere";

**return** **false**;

}

resultExpression[i][calcend[i + 1]] = -1 \* resultExpression[i + 1][calcend[i + 1]];

calcend[i] = calcend[i + 1];

calcbeg[calcend[i]] = i;

}

**else** **if** (!CalcOperation(**ref** i, **out** error)) **return** **false**;

}

}

**catch** (Exception e) { error= e.Message + " Expresie invalida::adunare & scadere"; **return** **false**; }

}

**return** **true**;

}

Bibliografie

Librarie pentru afisare notatii:

https://github.com/ForNeVeR/wpf-math