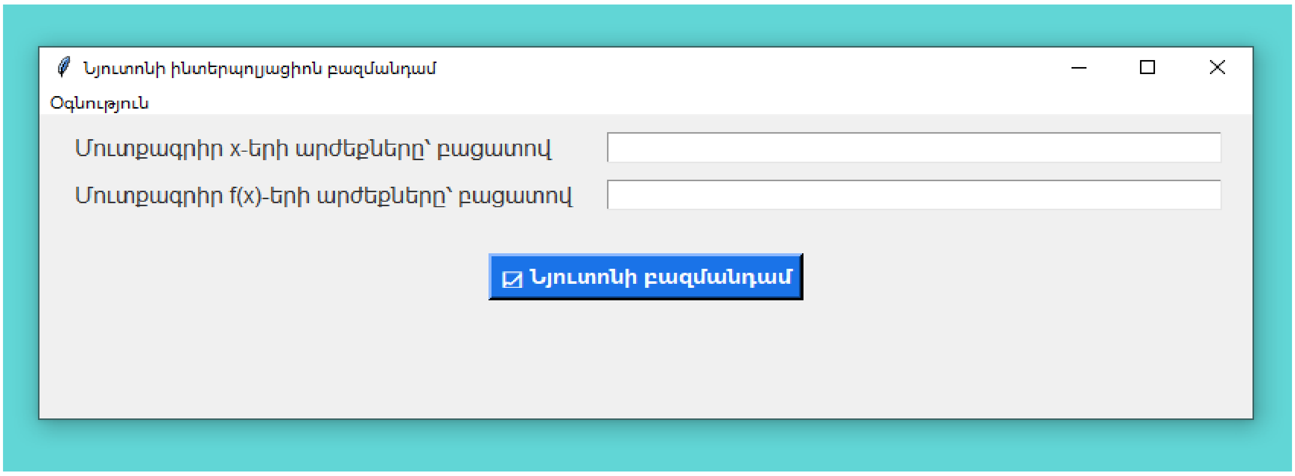
ԳԼՈՒԽ 4

Մոտարկման ալգորիթմների գործնական նշանակությունը

4.1 Թվային մոտարկման ալգորիթմի ծրագրային իրականացում Նյուտոնի բաժանված տարբերությունների մեթոդով

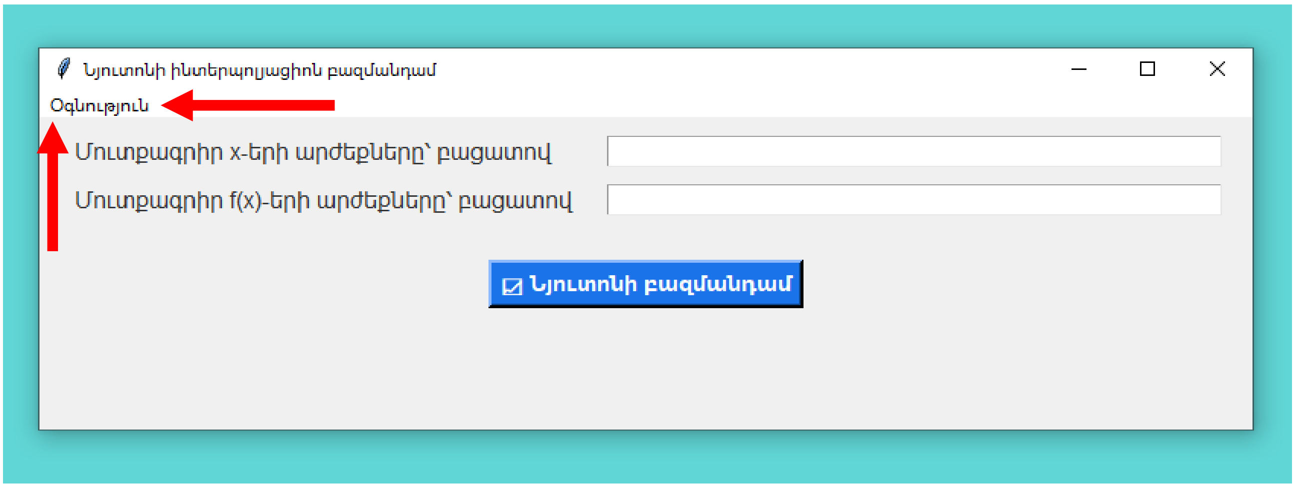
Շատ հաճախ, երբ առաջանում է բաժանված տարբերությունների աղյուսակի կառուցման և Նյուտոնի ինտերպոլյացիոն բազմանդամի ստացման անհրաժեշտությունը, ուսանողները, ինչպես նաև գիտական աշխատողները, փորձում են համացանցի միջոցով գտնել այնպիսի հարթակներ կամ կայքեր, որտեղ հնարավոր կլինի մուտքագրել համապատասխան տվյալները և ստանալ անհրաժեշտ արդյունքները: Այլընտրանքային մոտեցում է նաև Python կամ MATLAB ծրագրավորման լեզուների կիրառումը, որի միջոցով հնարավոր է ստանալ բաժանված տարբերությունները և ինտերպոլյացիոն բազմանդամը: Այս տարբերակներն իրենց բնույթով ունեն և՛ դրական, և՛ բացասական կողմեր: Դրականն այն է, որ նման գործընթացի արդյունքում օգտվողը ստիպված է լինում ուսումնասիրել բազմաթիվ աղբյուրներ, խորացնել թեմայի ըմբռնումը և հմտանալ թվային մոտարկման մեթոդներում: Սակայն բացասական կողմն այն է, որ ամբողջ գործընթացը կարող է խլել բավականին երկար ժամանակ, ինչը հատկապես խնդիր է, երբ անհրաժեշտ է արագ ստանալ արդյունք կամ տվյալ պահին ինտերնետ հասանելիություն չկա:

Այս խնդրի լուծման նպատակով մագիստրոսական թեզի շրջանակում իրականացվել է գործնական ծրագիր՝ .exe ֆայլի տեսքով, որը թույլ է տալիս պարզ և հասանելի կերպով, առանց լրացուցիչ գրադարանների ներբեռնման կամ ինտերնետային հարթակների դիմելու, մուտքագրել տվյալները և անմիջապես ստանալ բաժանված տարբերությունների աղյուսակն ու Նյուտոնի ինտերպոլյացիոն բազմանդամը: Ծրագիրը նախատեսված է ինչպես ուսանողների, այնպես էլ գիտական աշխատողների համար և հատկապես օգտակար է ուսումնական, հետազոտական և ինժեներական նպատակներով կիրառման ժամանակ: Այժմ ստորև կներկայացնեմ .exe ֆայլի օգտագործման հնարավորությունները և առավելությունները: Ֆայլն ունի հետևյալ տեսքը (նկ. 4.1).



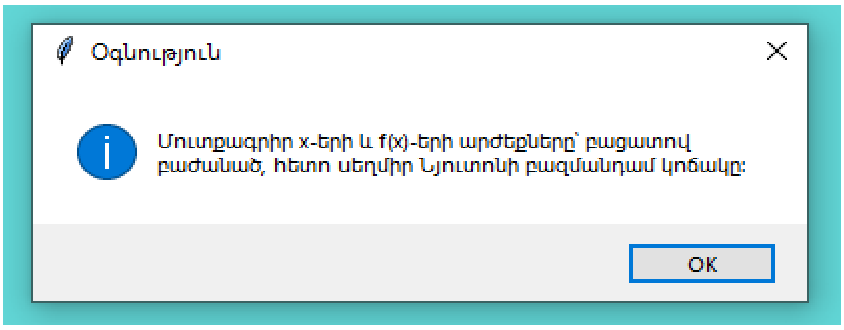
նկար 4.1

Ինչպես տեսնում ենք վերևի ձախ անկյունում առկա է «Օգնություն» կոճակը (նկ. 4.2).



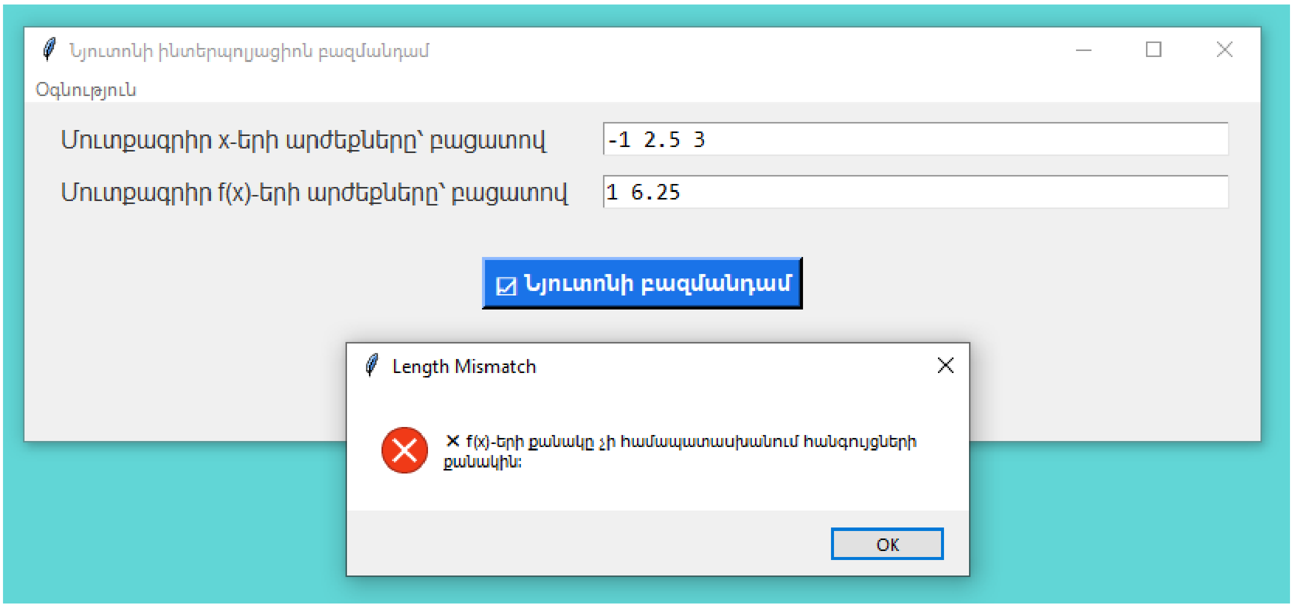
նկար 4.2

սեղմելով կոճակի վրա, բացվում է նոր պատուհան, որտեղ պարզ և համառոտ ներկայացված է ծրագրի աշխատելու համար անհրաժեշտ քայլերի հաջորդականությունը (նկ. 4.3).



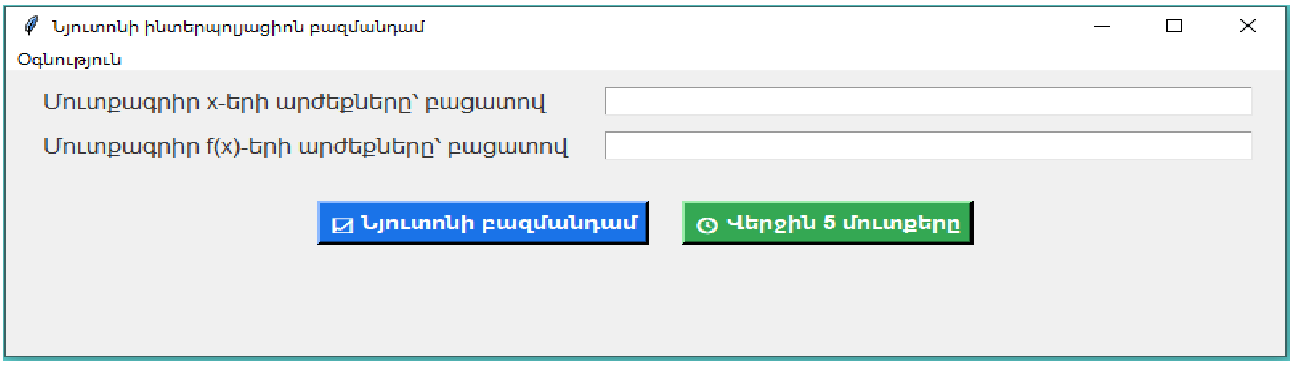
նկար 4.3

Այժմ հետևելով քայլերի հաջորդականությանը, մուտքագրենք պատահական x-եր և $f(x)$-եր, և տեսնենք ինչ հնարավորություններ ունի .exe ֆայլը: Դիտարկենք հետևյալ օրինակը (նկ. 4.4).



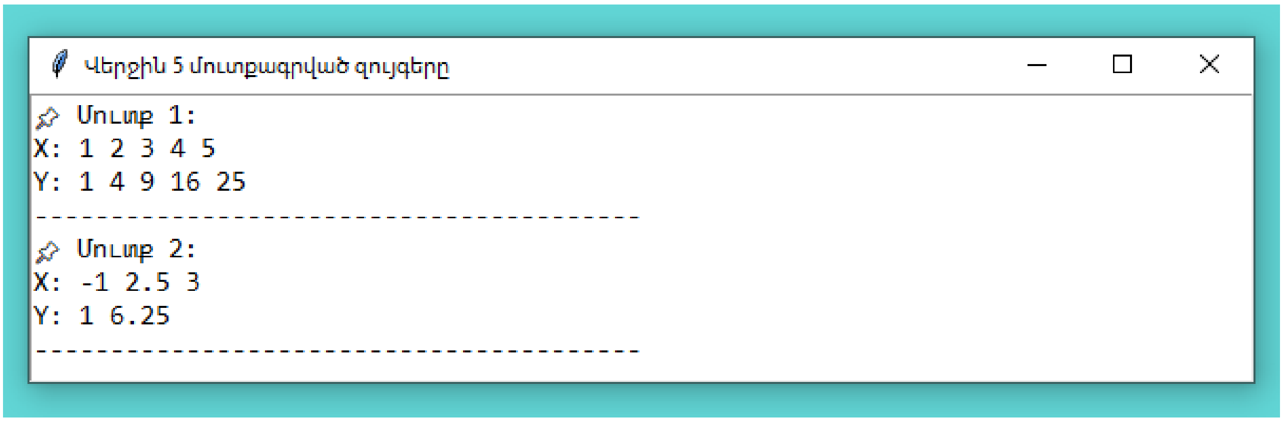
նկար 4.4

ինչպես տեսնում ենք մուտքագրելով ավելի շատ x-եր, քան $f(x)$-երն են արդյունքում ստանում ենք հաղորդագրություն որ $f(x)$-երի քանակը չի համապատասխանում հանգույցների քանակին: Նույն հաղորդագրությունը կստանանք նաև երբ $f(x)$-երի քանակը գերազանցի հանգույցների քանակին: Սեղմելով OK կոճակին բոլոր մուտքագրված արժեքները կջնջվեն: Սակայն դա անհարմարություն է պատճառում օգտատիրոջը: Օգտատերը ստիպված պետք է նորից մուտքագրի արժեքները, և չի բացառվում որ կարող է նորից սխալ լրացնել մուտքագրման համար անհրաժեշտ դաշտերը: Այդ պատճառով կարող ենք ավելացնել հետևյալ ֆունկցիան (կոճակը) (նկ. 4.5).



նկար 4.5

Սեղմելով «Վերջին 5 մուտքրեր» կոճակի վրա կբացվի նոր պատուհան (նկ. 4.6).

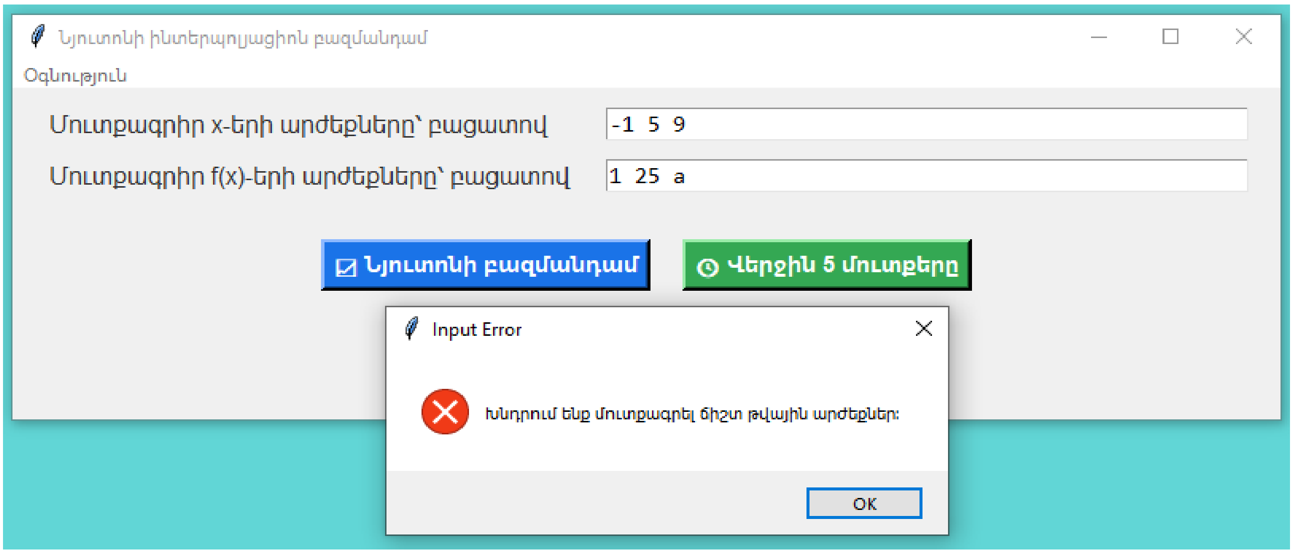


նկար 4.6

Ինչպես տեսնում ենք բացված պատուհանում ցույց է տալիս 2 մուտքերը, եթե դրանք գերազանցեն 5-ին, ապա ցույց կտա վերջին 5 մուտքերը, իսկ եթե բացակայի մուտքերը, այսինքն ընդհանրապես թվեր մուտքագրված չլինեն, ապա օգտատերը կտեսնի հետևյալ պատուհանը (նկ. 4.7).

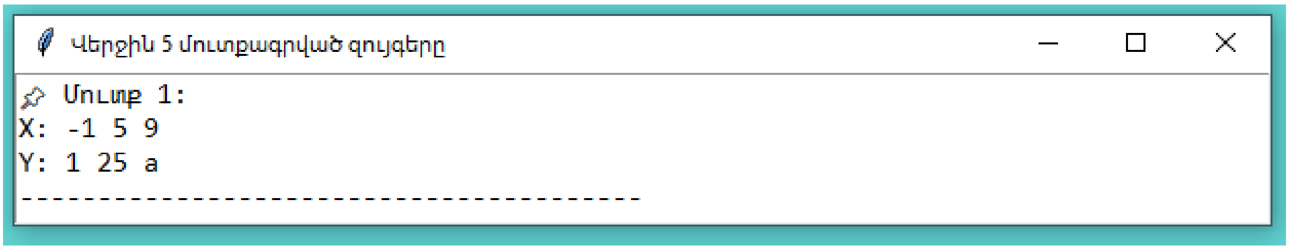
նկար 4.7

Այժմ դիտարկենք այն տարբերակը երբ օգտատերը պատահաբար թվային արժեք մուտքագրելու փոխարեն մուտքագրել է սիմվոլային արժեք: Նման պարագայում նույնպես անհրաժեշտ է զգուշացնող պատուհան սխալի վերաբերյալ (նկ. 4.8).



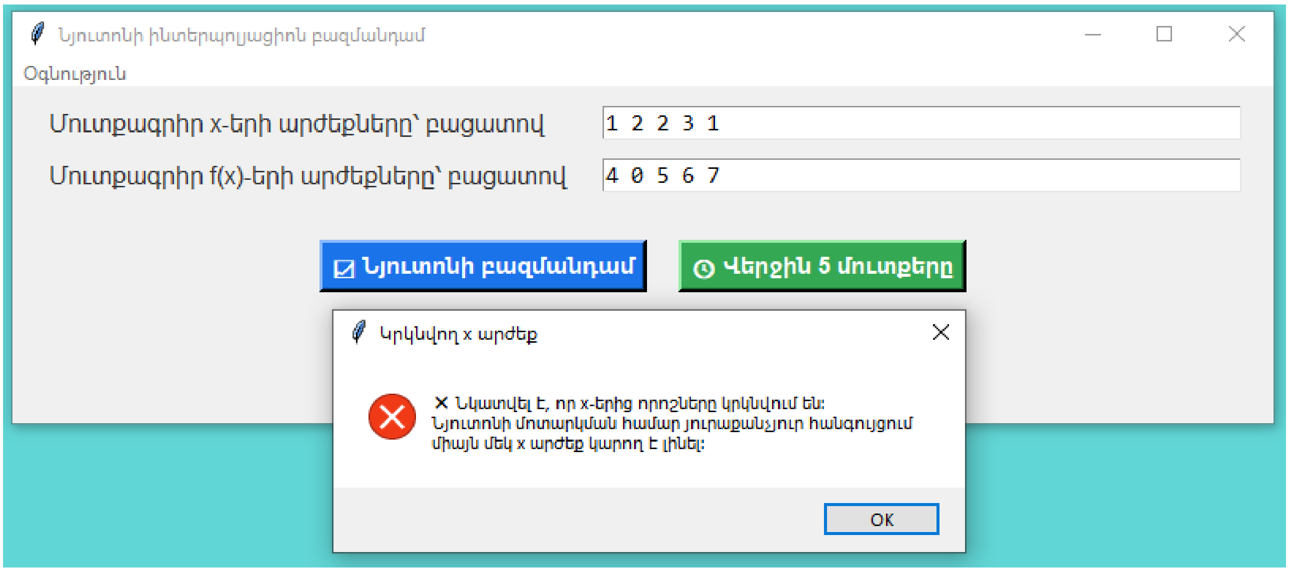
նկար 4.7

Այսինքն ստացվեց որ ցանկացած սիմվոլային արժեք մուտքագրելու դեպքում կստանանք տվյալ զգուշացնող հաղորդագրությունը: Իդեպ, այս դեպքում նույնպես մուտքագրված արժեքները պահպանվում են (նկ. 4.9).



նկար 4.9

Այժմ դիտարկենք այն տարբերակը երբ օգտատերը պատահաբար մուտքագրել է այնպիսի x-եր, որոնց մեջ կան կրկնություններ: Ինչպես գիտենք դա սխալ է, այսինքն հնարավոր չէ միևնույն x կետում ֆունկցիան ընդունի տարբեր $f(x)$ արժեքներ: Հետևաբար հարկավոր է ևս մեկ զգուշացնող պատուհան ստեղծել նման խնդրի համար (նկ. 4.10).

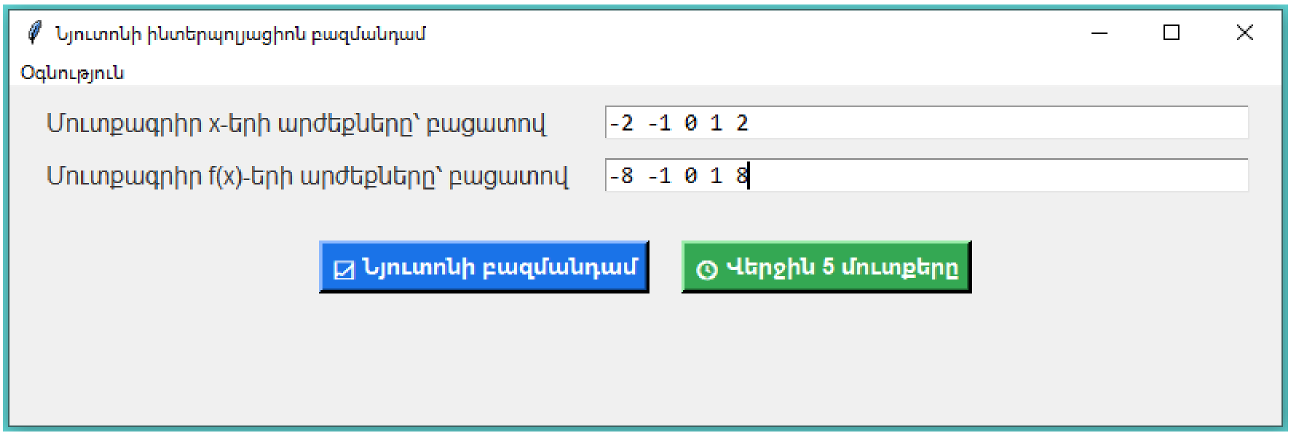


նկար 4.10

Այսպիսով, ստացվեց որ ցանկացած սխալ մուտքագրման դեպքում, միևնույն է ծրագիրը չի աշխատելու, այսինքն չի կազմելու բաժանված տարբերությունների աղյուսակը, ինչպես նաև Նյուտոնի ինտերպոլյացիոն բազմանդամը: Այժմ դիտարկենք այն դեպքը երբ բոլոր մուտքագրված տվյալները ճիշտ են, այսինքն բոլոր վերը թվարկված օրինակները բացառում ենք: Ենթադրենք ունենք հետևյալ տվյալները (Աղյուսակ 4.1).

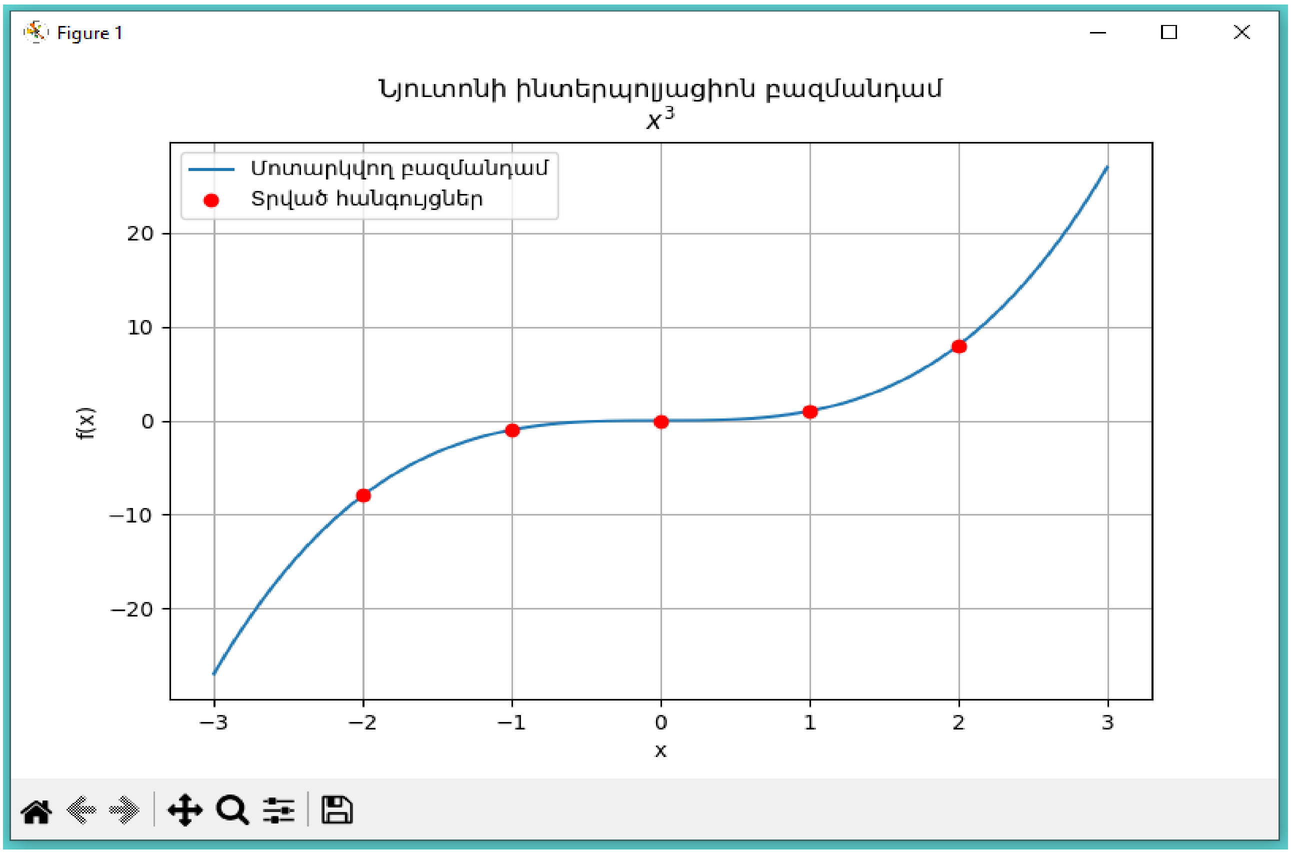
Աղյուսակ 4.1

x	-2	-1	0	1	2
$f(x)$	-8	-1	0	1	8

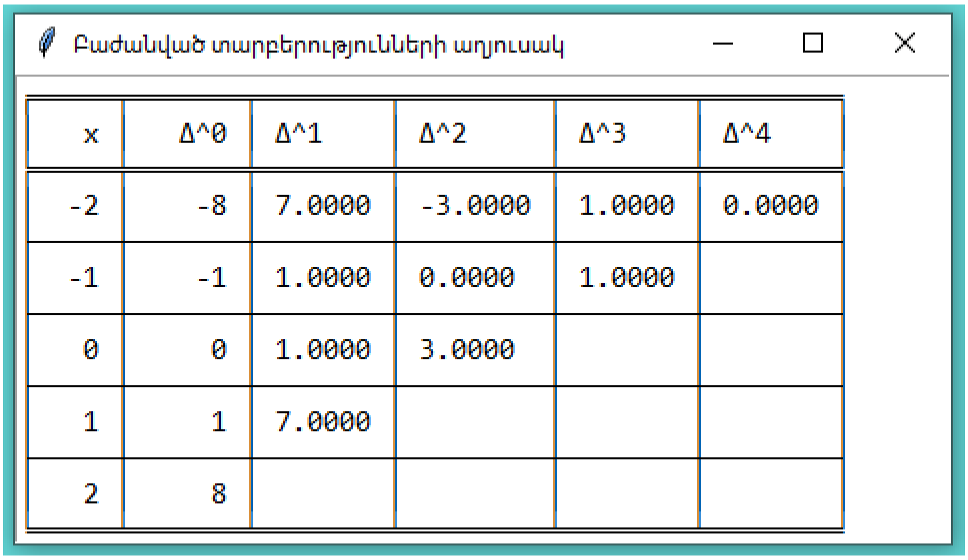


նկար 4.11

եթե մուտքագրենք ծրագրում, և սեղմենք «Նյուտոնի բազմանդամ» կոճակի վրա, կստանանք (նկ. 4.12, և նկ. 4.13).



նկար 4.12

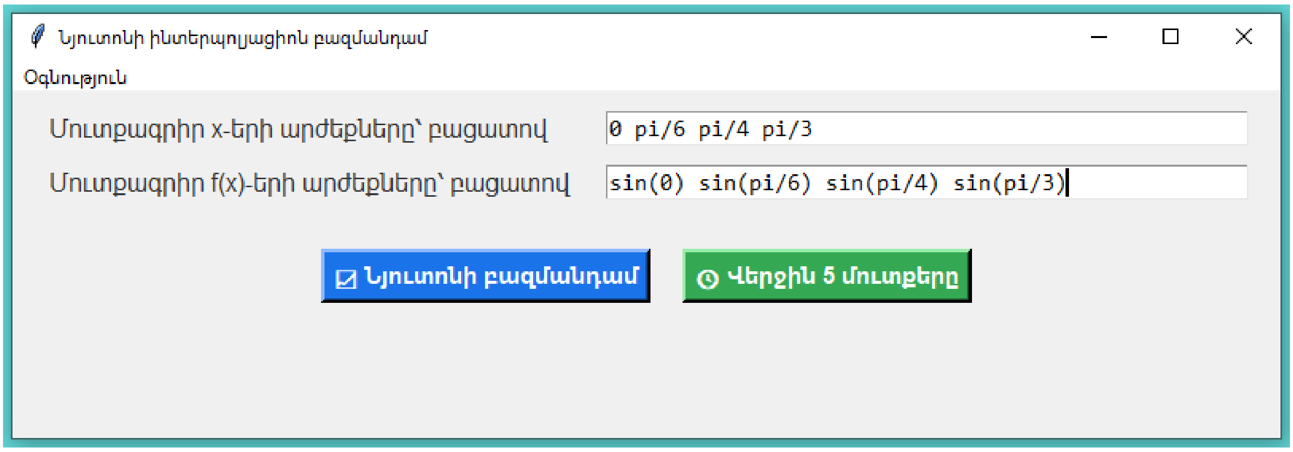


x	Δ^0	Δ^1	Δ^2	Δ^3	Δ^4
-2	-8	7.0000	-3.0000	1.0000	0.0000
-1	-1	1.0000	0.0000	1.0000	
0	0	1.0000	3.0000		
1	1	7.0000			
2	8				

նկար 4.13

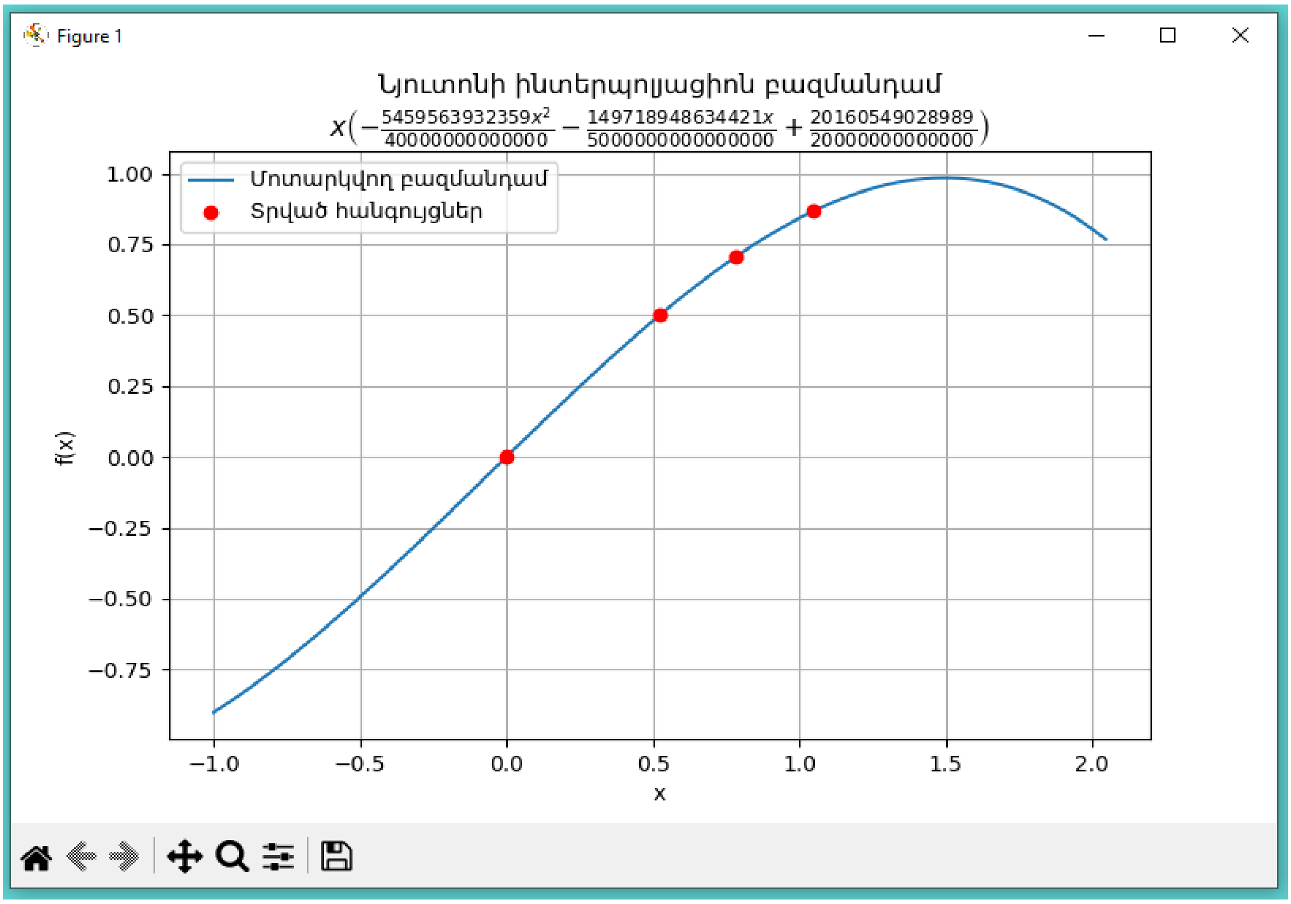
Այսպիսով ստացանք և՛ բաժանված տարբերությունների աղյուսակը, և՛ Նյուտոնի ինտերպոլյացիոն բազմանդամը, ինչը և պահանջվում էր գտնել: (Նկ. 4.12)-ի ներքևի ձախ մասում առկա են տարբեր ծրագրային գործիքներ, որոնց միջոցով հնարավոր է գրաֆիկը որպես նկար ներբեռնել, մեծացնել/փոքրացնել գրաֆիկի չափերը, վերադառնալ սկզբնական տեսքին և այլն:

Թվում է թե, այս օրինակը բավարար է պնդելու համար, որ ծրագիրը անթերի է աշխատում բոլոր ճիշտ մուտքագրված թվային արժեքների դեպքում: Այժմ հարց է առաջանում, ինչ արդյունք կստանանք, եթե մեզ տրված արժեքները լինեն եռանկյունաչափական կամ լոգարիթմական տեսքի: Դիտարկենք հետևյալ եռանկյունաչափական օրինակը (նկ. 4.14).



նկար 4.14

սեղմելով «Նյուտոնի բազմանդամ» կոճակի վրա, կստանանք հետևյալը (նկ. 4.15 և նկ. 4.16).



նկար 4.15

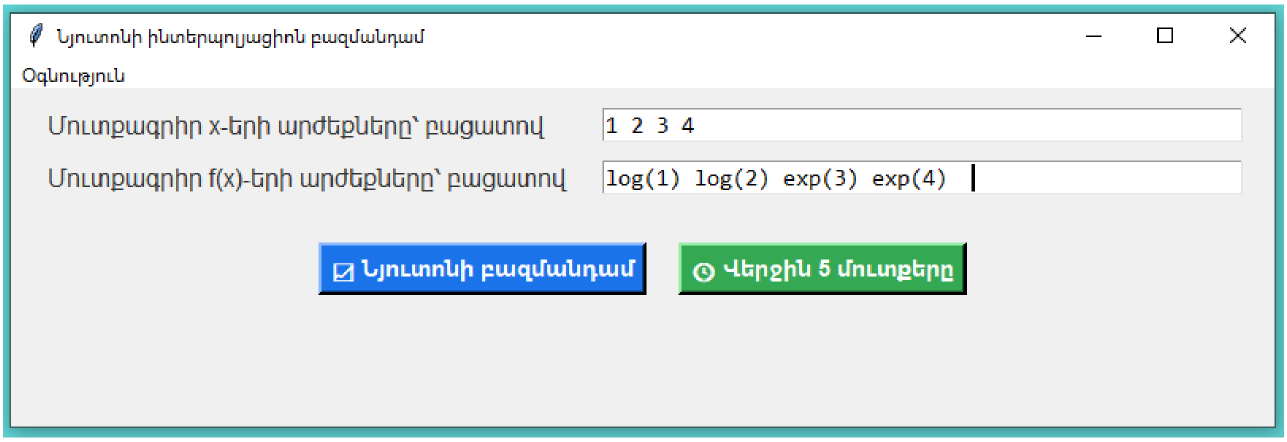
Բաժանված տարբերությունների աղյուսակ

x	Δ^0	Δ^1	Δ^2	Δ^3
0	0	0.9549	-0.2086	-0.1365
0.5236	0.5	0.7911	-0.3515	
0.7854	0.7071	0.6070		
1.0472	0.866			

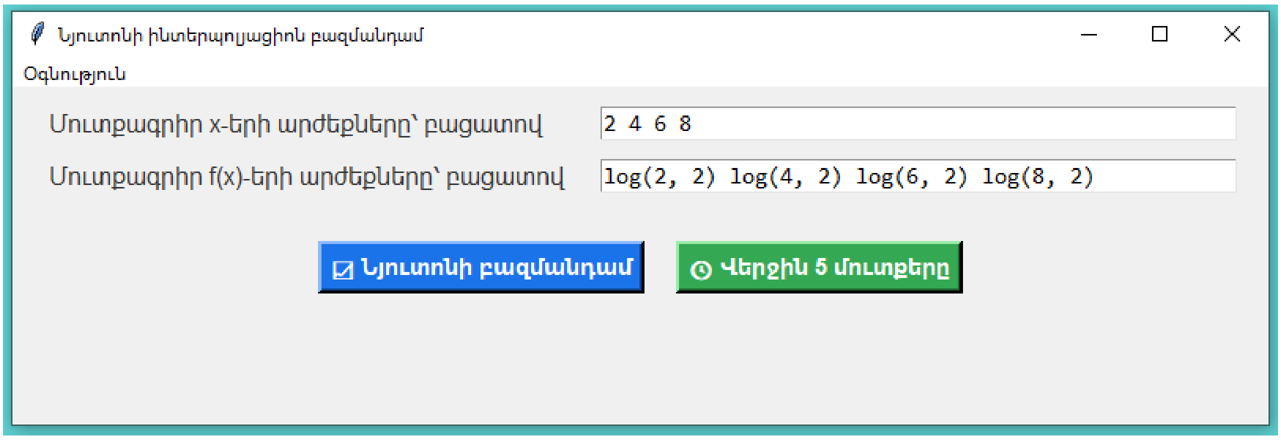
նկար 4.16

Ինչպես տեսնում ենք, այս դեպքում ևս առանց որևէ խնդիր ծրագիրը անթերի աշխատեց: Այժմ դիտարկենք լոգարիթմական օրինակներ.

1) Լոգարիթմ բնական հիմքով (e հիմքով) (նկ. 4.17)
2) Լոգարիթմ առանց բնական հիմքի (նկ. 4.18)

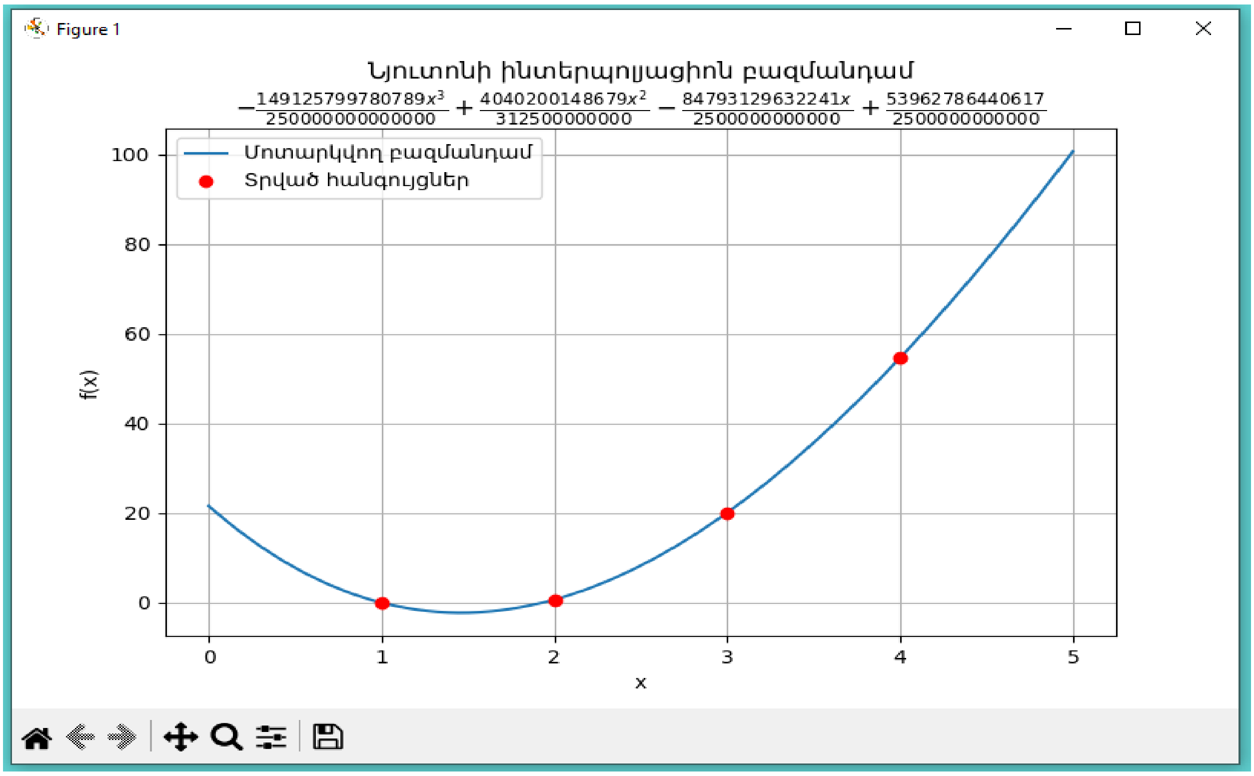


նկար 4.17



նկար 4.18

Առաջին օրինակի դեպքում ստանում ենք հետևյալ արդյունքները (նկ. 4.19, և նկ. 4.20)



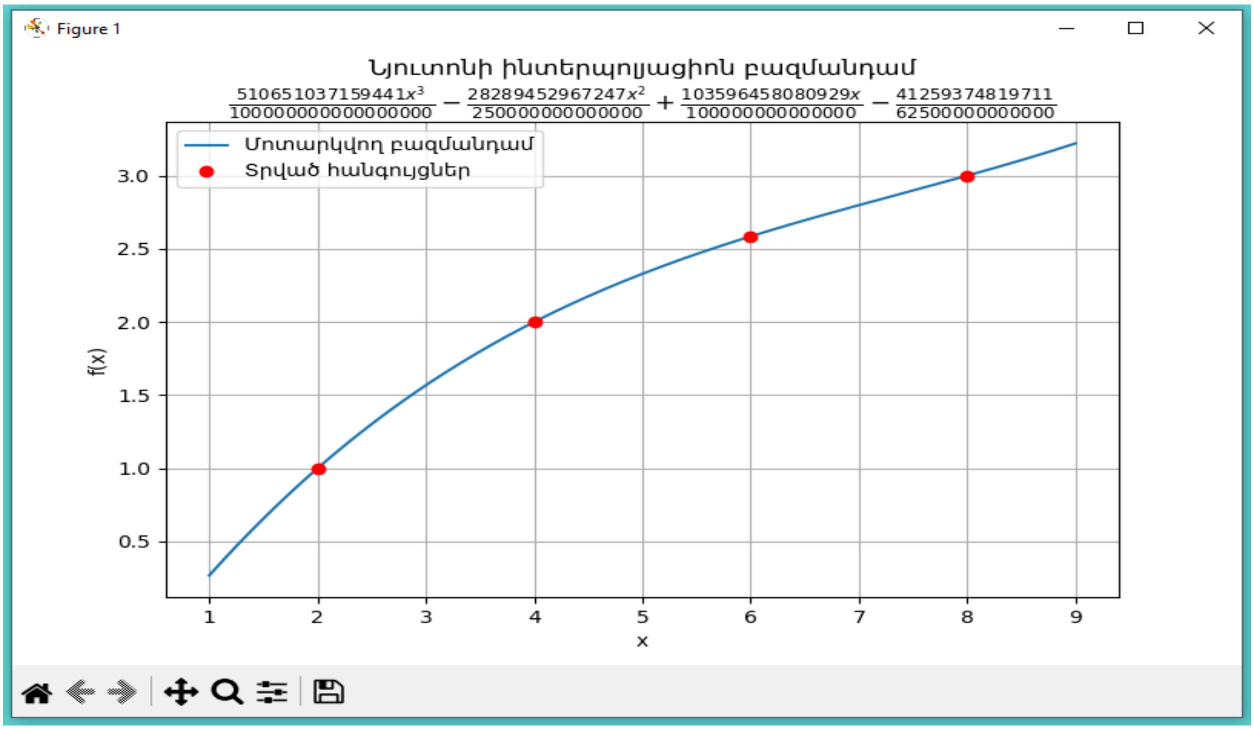
նկար 4.19

Բաժանված տարբերությունների աղյուսակ

x	Δ^0	Δ^1	Δ^2	Δ^3
1	0	0.6931	9.3496	-0.5965
2	0.6931	19.3924	7.5601	
3	20.0855	34.5126		
4	54.5982			

նկար 4.20

Երկրորդ օրինակի դեպքում ստանում ենք հետևյալ արդյունքները (նկ. 4.21, և նկ. 4.22)



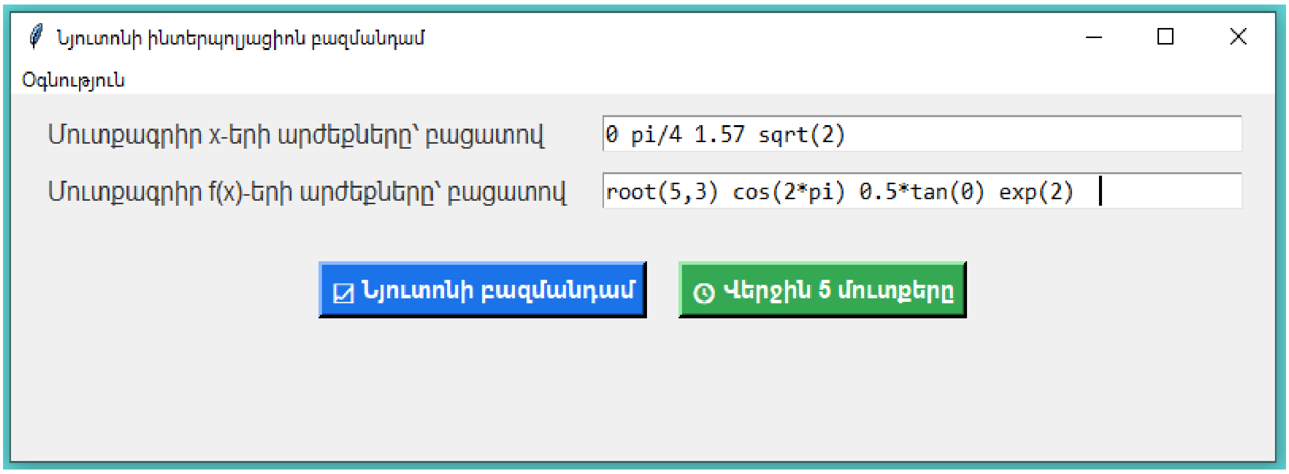
նկար 4.21

Բաժանված տարբերությունների աղյուսակ

x	Δ^0	Δ^1	Δ^2	Δ^3
2	1	0.5000	-0.0519	0.0051
4	2	0.2925	-0.0212	
6	2.585	0.2075		
8	3			

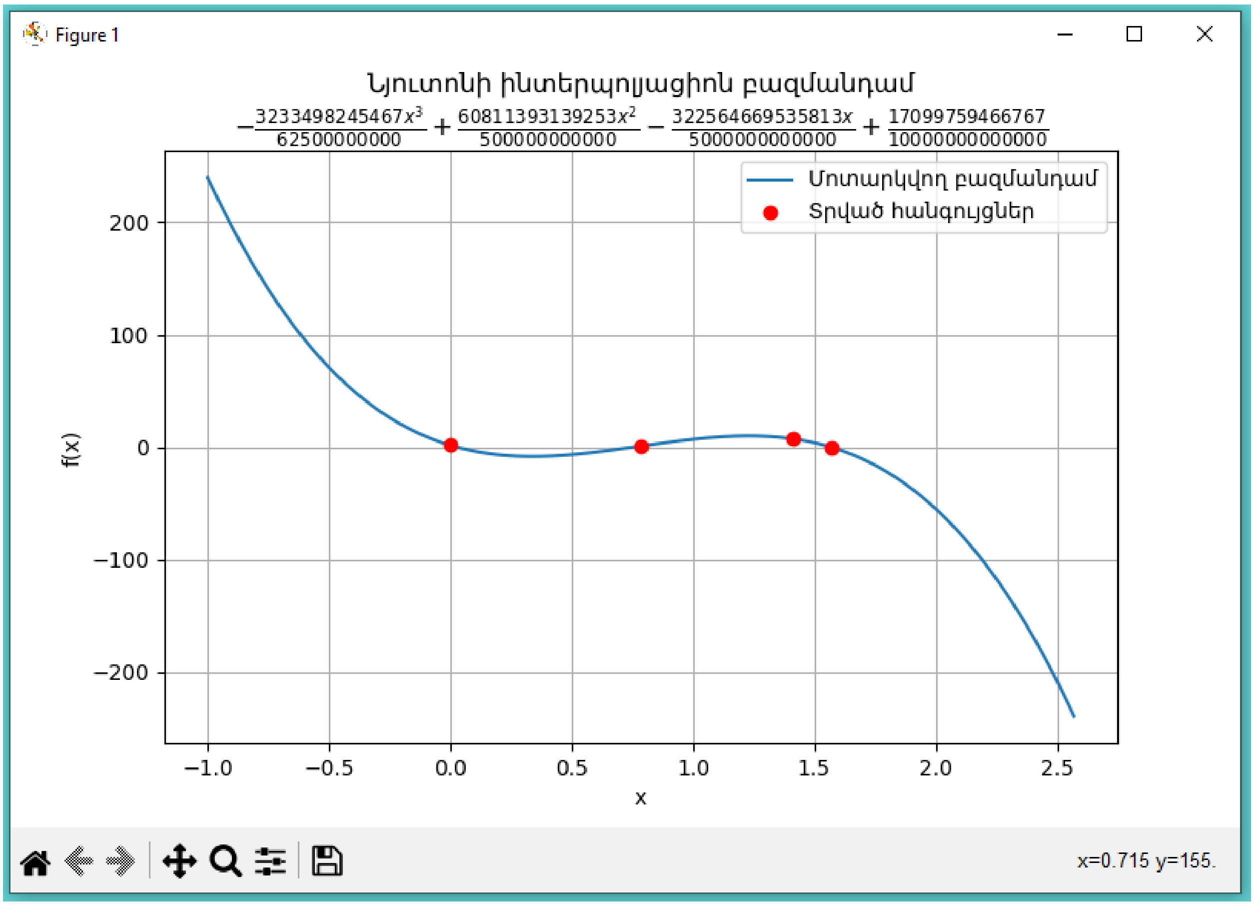
նկար 4.22

Այսպիսով, ստացվեց որ և՛ լոգարիթմական , և՛ եռանկյունաչափական տեսքով տրված արժեքների դեպքում ծրագիրը անթերի աշխատում է: Այժմ դիտարկենք վերջին օրինակը, որը իր մեջ ներառում է միաժամանակ բոլոր հնարավոր տարբերակներով տրված արժեքներ (նկ. 4.23).



նկար 4.23

Այս դեպքում կստանաք հետևյալ արդյունքները (նկ. 4.24 և նկ. 4.25).



նկար 4.24

Բաժանված տարբերությունների աղյուսակ

x	Δ^0	Δ^1	Δ^2	Δ^3
0	1.71	-0.9040	-0.2360	-51.7360
0.7854	1	-1.2745	-73.4017	
1.57	0	-47.4307		
1.4142	7.3891			

Նկար 4.25

Թվային մոտարկման խնդիրները շատ կարևոր դեր են խաղում ոչ միայն կիրառական մաթեմատիկայի, այլ նաև տեխնիկական գիտությունների ոլորտում: Այս աշխատանքում ներկայացված ծրագիրը, իրենից ներկայացնում է գործնական պարզ լուծում: Հաշվի առնելով, որ իրական խնդիրներում հաճախ անհրաժեշտ է աշխատել ոչ միայն թվային, այլև ֆունկցիոնալ արտահայտությունների (օրինակ՝ sin(pi/4), log(8, 2), exp(1) և այլն) հետ, ստեղծվել է Python լեզվով աշխատող .exe ֆայլ, որը թույլ է տալիս մուտքագրել տարբեր տեսքի արտահայտություններ, ստանալ բաժանված տարբերությունների աղյուսակը, կառուցել Նյուտոնի ինտերպոլյացիոն բազմանդամը և պատկերել այն գրաֆիկորեն: Ծրագիրը պարունակում է նաև վերջին մուտքագրումների պահպանման և դիտման հնարավորություն, ինչը բարձրացնում է օգտատիրոջ հարմարավետությունը և թույլ է տալիս վերադարձ կատարել նախկին տվյալներին, առանց կրկնակի մուտքագրման անհրաժեշտության: Բացի այդ, ծրագիրը տրամադրում է սխալների հայտնաբերման պատուհաններ, զգուշացնելով օգտատիրոջը հնարավոր մուտքային անհամապատասխանությունների մասին: Այն հնարավորություն է տալիս նաև ընդլայնել գործառույթները, ներառելով նոր մոտարկման մեթոդներ, սխալի վերլուծություն, տվյալների ներմուծման/արտահանման ֆունկցիաներ, ինչպես նաև մշակել վեբ կամ մոբայլ տարբերակներ: