Šolski center Celje
Srednja šola za kemijo, elektrotehniko in računalništvo

PROGRAM ZA DELO S FUNKCIJAMI

raziskovalna naloga

PODROČJE: Računalništvo

Avtorica:
Maša KRANER, R-4.a

Mentor:
mag. Boštjan RESINOVIČ, univ. dipl. inž. rač.

Mestna občina Celje, Mladi za Celje
Celje, 2021

Povzetek

Cilj raziskovalne naloge je bila nadgradnja aplikacije ProGrafi, ki sem jo naredila v sklopu lanske raziskovalne naloge. Želela sem ustvariti medpredmetno povezavo med matematiko in programiranjem. Nadgrajena aplikacija ProGrafi 1.1 zna izrisovati vse elementarne funkcije, torej linearno, kvadratno, potenčno, korensko, eksponentno, logaritemsko, polinomsko in racionalno funkcijo ter vsebuje različne tipske naloge, ki pomagajo dijakom pri lažjem razumevanju le-teh. Aplikacija uporablja programsko platformo Windows Forms Application, napisana je v Visual Studiu s pomočjo bližnjice ZedGraph, ki omogoča risanje grafov.

KLJUČNE BESEDE: matematične elementarne funkcije, ZedGraph, Windows Forms Application, Visual studio, matematika, programiranje.

Abstract

The aim of the research task was to upgrade the ProGrafi applications, which was created as part of last year's research tasks. I wanted to create a connection between mathematics and computer science. The upgraded application ProGrafi 1.1 is able to plot all elementary functions, so it contains linear, quadratic, potential, root, exponential, logarithmic, polynomial and rational functions, and contains various type exercises to help students more easily understand mentioned functions. The application uses the Windows Forms Application software platform, written in Visual Studio with the help of the ZedGraph class library, which allows you to draw graphs.

KEY WORDS: mathematical elementary functions, ZedGraph, Windows Forms Application, Visual studio, mathematics, programming.

KAZALO VSEBINE

KAZALO SLIK

KAZALO GRAFOV

ZAHVALA

Pri izdelavi raziskovalne naloge se zahvaljujem svojemu mentorju, mag. Boštjanu Resinoviču, za pomoč pri snovanju naloge ter reševanje problemov z vidika programiranja. Zahvaljujem se tudi prof. Nataši Besednjak za pomoč pri reševanju matematičnih problemov.

Zahvaljujem se tudi lektorici, prof. Tjaši Verdev.

1 UVOD

Namen raziskovalne naloge je nadgradnja aplikacije ProGrafi, ki sem jo naredila v sklopu lanske raziskovalne naloge. ProGrafi sem ustvarila z namenom, da bi povezala dva predmeta, ki me najbolj zanimata – matematiko in programiranje. V nadaljevanju bom najprej opisala programe, ki sem jih uporabljala za izdelavo aplikacije, in sicer Visual Studio, Winodows Forms Application in ZedGraph. Na kratko bom opisala tudi vse zajete matematične elementarne funkcije, tako tiste, ki sem jih opisala že v lanski raziskovalni nalogi, kot tiste, ki sem jih letos dodala k nalogi. Tako bodo opisane, poleg linearne in kvadratne funkcije, še potenčna funkcija, korenska funkcija, eksponentna funkcija, logaritemska funkcija in polinomska ter racionalna funkcija. V aplikacijo ProGrafi sem dodala tudi razne tipske naloge pri vsaki funkciji, ki so v nadaljevanju prav tako podrobneje opisane.

Pri raziskovanju sem si pomagala z anketo, katere izsledke bom prav tako predstavila. Rezultati ankete so mi pomagali pri oblikovanju aplikacije in sestavljanju tipskih nalog pri vsaki funkciji. Nova verzija aplikacije se tako imenuje ProGrafi 1.1. Podrobneje bom opisala tudi nadgradnjo omenjene aplikacije, njen namen, kaj zajema nadgradnja in njeno uporabnost.

1.1 Predstavitev problema

V času šolanja sem se velikokrat spopadla s težavo, da nikjer nisem našla zbranih vseh podatkov o funkcijah, nikjer ni bilo rešenih tipskih nalog, kot jih rešujemo v šoli, ipd. To je bil povod, da sem se odločila pripraviti takšno aplikacijo, ki bo pomagala vsem srednješolcem pri reševanju matematičnih funkcij.

1.2 Hipoteze

Preden sem začela z raziskovanjem sem postavila naslednjih 5 hipotez:

H1: Matematični del bo zame predstavljal večjo težavo kot programski.

H2: Anketa bo pripomogla pri nadgradnji programa.

H3: Srednješolsko znanje ne zadostuje za izdelavo aplikacije.

H4: Anketiranci bodo za najtežjo funkcijo izbrali racionalno funkcijo.

H5: Aplikacija bo risala vse elementarne funkcije, ki se obravnavajo v srednji šoli.

1.3 Raziskovalne metode

Moje raziskovanje se je začelo z anketiranjem in zraven še z zbiranjem informacij, ki so mi kasneje pomagale pri izdelavi celotne aplikacije. Na začetku sem se usmerila k matematičnemu delu, kjer sem naredila seznam vseh elementarnih funkcij, ki se obravnavajo pri pouku matematike v srednji šoli.

Kot sem že omenila, sem za raziskovanje uporabila tudi metodo anketiranja, s katero sem ugotovila, katere matematične funkcije predstavljajo dijakom največje težave. Po pripravljenem seznamu sem se lotila programskega dela. Samo osnovo za risanje sem imela že od lanske raziskovalne naloge, tako da sem morala vse nadgraditi, dodelala sem tudi izgled aplikacije in njene funkcionalnosti. Aplikacijo sem z izgledom in uporabnostjo želela približati čim večjemu krogu uporabnikov in jih s tem spodbuditi k zanimanju za matematiko. Pomagala sem si predvsem z že narejenimi aplikacijami, kot so na primer Graph, Desmos, PhotoMath, ipd.

1.4 Cilji

Glavni cilj raziskovalne naloge je nadgraditi aplikacijo ProGrafi za delo s funkcijami, ki bo povezovala predmetni področji matematike in programiranja. Aplikacija ProGrafi 1.1 bo risala in računala več elementarnih funkcij, kot jih je aplikacija ProGrafi Cilj je nadgraditi aplikacijo do te mere, da bo zajemala vse elementarne funkcije, ki se obravnavajo pri predmetu matematika v srednjih šolah in jo narediti uporabno za pomoč pri učenju.

.

2 UPORABLJENA ORODJA IN PLATFORME

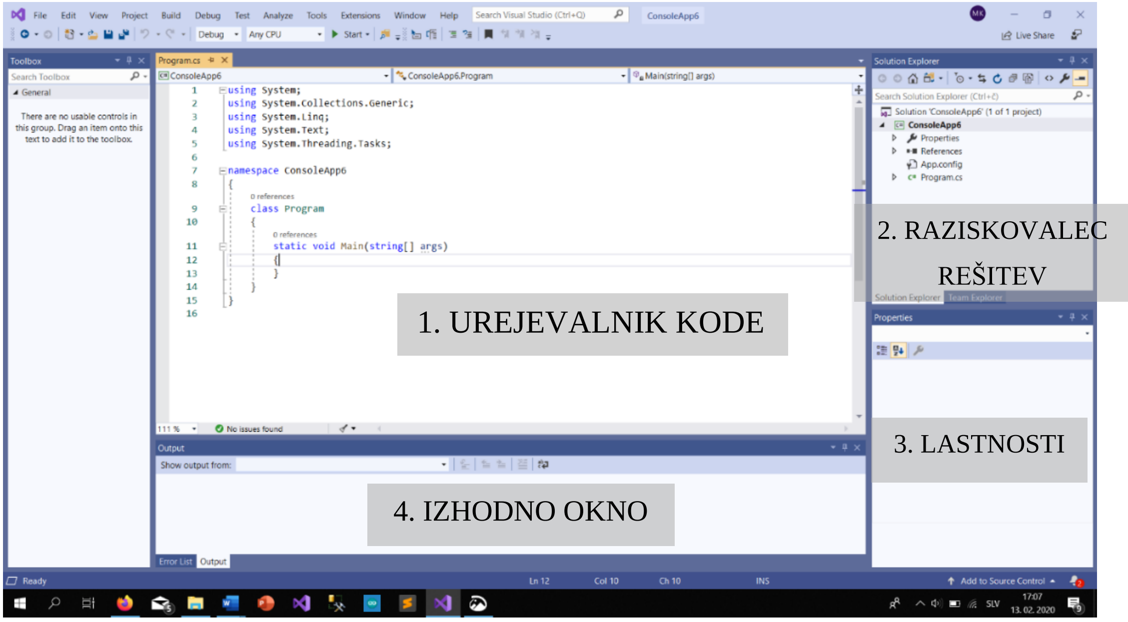
2.1 Visual Studio

Visual Studio je integrirano razvojno okolje (IDE). Uporablja se predvsem za pisanje računalniških programov, spletnih mest, spletnih storitev in spletnih aplikacij. Večina razvojnih okolij ponuja samo standardne urejevalnike in razhroščevalnike. Visual Studio poleg tega nudi še prevajalnike, orodja za dokončanje kode, grafične oblikovalce in številne druge funkcije, ki uporabniku olajšajo postopek razvoja programske opreme. Podpira več programskih jezikov (C, C++, C«, Python, HTML/CSS) in je na voljo v brezplačni različici »Community« ter v plačani komercialni različici »Professional«.

Različica Community vsakemu posamezniku omogoča, da razvije svoje brezplačne ali plačljive spletne aplikacije. To različico lahko skupaj uporablja do pet uporabnikov, njen glavni namen pa je zagotoviti večji dostop do razširitev in omogočiti urejanje čim več različnih programskih jezikov. Različica Professional ponuja podporo za urejanje XML in XSLT ter vključuje orodje, kot je raziskovalec strežnikov in integracijo s strežnikom Microsoft SQL. Na voljo je tudi brezplačna poskusna različica te izdaje, ki jo mora uporabnik po poskusnem obdobju plačati, da jo lahko še naprej uporablja. Njeni glavni namen je zagotoviti fleksibilnost (profesionalna orodja za gradnjo katere koli vrste aplikacij), storilnost in sodelovanje med uporabniki.

Preden se uporabnik loti pisanja kode, se mu prikaže začetno okno, ki ga lahko vidite na Sliki 1. Na njej so označena glavna okna in njihova orodja, ki so v nadaljevanju opisana.

1. **UREJEVALNIK KODE** (workspace) – mesto, kamor uporabnik piše svojo programsko kodo.
2. **RAZISKOVALEC REŠITEV**(solution explorer) – prikaže datoteke, s katerimi uporabnik trenutno dela.
3. **LASTNOSTI** (properties) – poda dodatne informacijo o izbranih delih trenutnega projekta.
4. **IZHODNO OKNO** (output) – prikazuje izhode, opozorila prevajalnika, sporočila o napakah in informacije za odpravljanje napak.



Slika 1: Glavno okno Visual Studio

2.2 Windows Forms Application

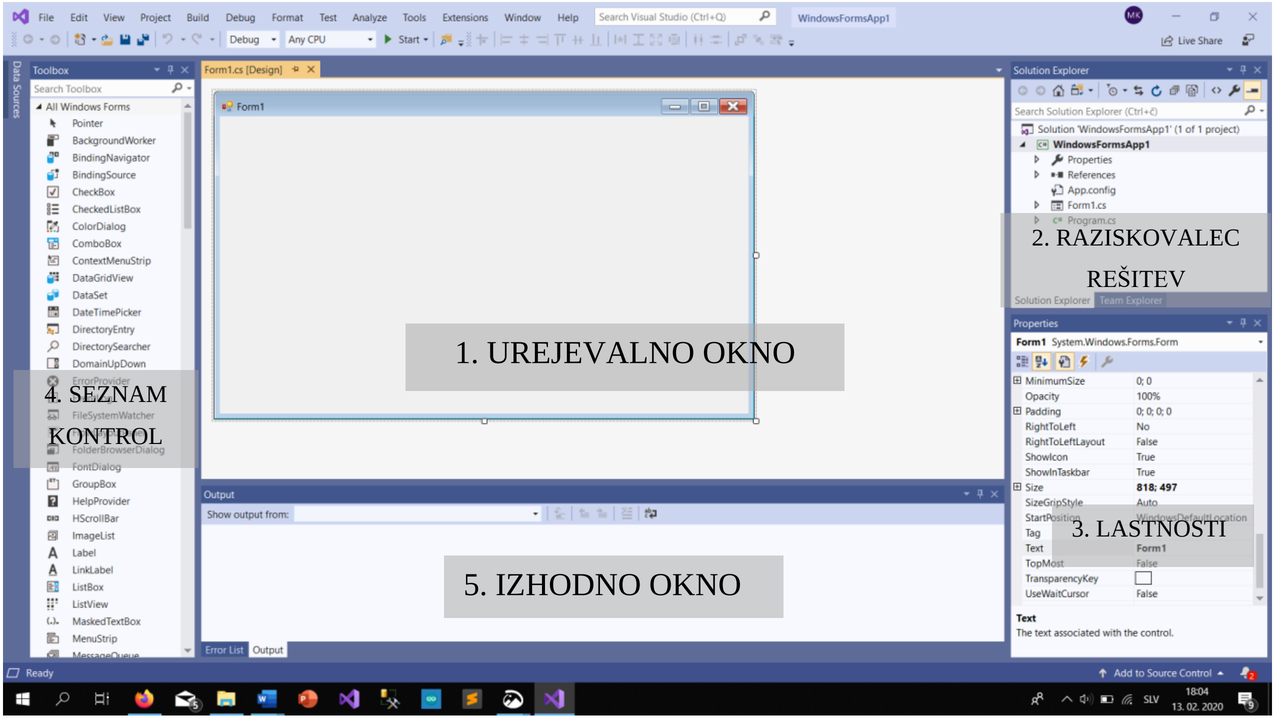
Windows Form Application je programska platforma s knjižnico razredov grafičnega uporabniškega vmesnika (GUI), ki je združena v .Net Framework. Njegov glavni namen je zagotoviti lažji vmesnik za razvoj grafičnih aplikacij za namizne, tablične in osebne računalnike. Imenujejo ga tudi WinForms. Zasnovan je za izdelovanje namiznih aplikacij, ki ne potrebujejo dostopa do spleta. Ponuja različne kontrole, kot so besedilna polja, gumbi, vstavljanje povezav do spletnih strani, možnosti za ustvarjanje kontrol po meri, itd. Tabelarični podatki, ki so vezani na XML, bazo podatkov itd., se lahko prikažejo s pomočjo nadzora DataGrid View v obliki vrstic in celic. Vsebuje glavno okno, kamor lahko te kontrolnike razporedimo kot sami želimo. Z možnostjo dodajanja kode v enem od kontrol se ustvari dogodek. Aplikacija na te dogodke reagira s pomočjo kode in obdeluje dogodke, ko se pojavijo.

Preden začne uporabnik razvijati določeno aplikacijo, se mu pojavi začetno okno, ki ga lahko vidite na Sliki 2. Na njej so označena glavna okna in orodja, ki so opisana v nadaljevanju.

1. **UREJEVALNO OKNO** (editor window) – mesto, kamor postavljamo razne kontrolnike, ki jih vidimo na levi strani programa (Toolbox). Na začetku je v urejevalnem oknu postavljeno samo eno prazno okno, na katerega lahko dvakrat kliknemo in pojavi se okno s kodo.
2. **RAZISKOVALEC REŠITEV** (solution explorer) – prikazane so lastnosti določenega kontrolnika, ko nanj kliknemo.

.

3. **LASTNOSTI** (properties) – uporablja se za spreminjanje različnih lastnosti izbranega elementa. Prav tako lahko spremenimo lastnosti komponent oz. kontrolnikov.
4. **SEZNAM KONTROL** (toolbox) – prikazuje seznam vseh kontrol, ki jih Windows Forms Application vsebuje.
5. **IZHODNO OKNO** (output) – prikazuje izhode, opozorila prevajalnika, sporočila o napakah in informacije za odpravljanje napak.



Slika 2: Glavno okno Windows Forms Application

2.3 ZedGraph

ZedGraph je knjižnica razredov, napisanih v programskem jeziku C#, ki je dostopna spletnemu nadzoru. Knjižnica je namenjena ustvarjanju različnih 2D grafov, navpičnih ali vodoravnih grafikonov, črtnih grafikonov in tortnih grafov. Njegova slabost je, da še ne ustvarja 3D površine ali grafikona. Ustvarjeni grafikoni so lahko sestavljeni z naslovi osi, legendo in puščicami. Ti razredi zagotavljajo visoko stopnjo prilagodljivosti, saj je skoraj vsak graf mogoče uporabniško spremeniti. ZedGraph prav tako vključuje tudi vmesnik UserControl, ki omogoča urejanje »povleci in spusti« v urejevalniku VisualStudio. Do ZedGrapha lahko dostopamo na spletni strani https://sourceforge.net/projects/zedgraph/files/, na kateri namestimo tisto različico knjižnice, ki se nam zdi najuporabnejša.

.

3 ELEMENTARNE MATEMATIČNE FUNKCIJE

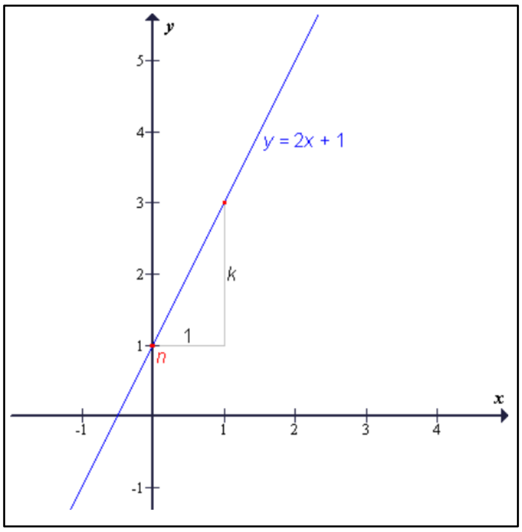
Funkcija f iz množice A v množico B je predpis, ki vsakemu elementu χ množice A priredi natanko določen element γ množice B. Elemente v množici A imenujemo originali, v množici B pa funkcijske vrednosti originalov imenujemo slike. Funkcijsko zvezo lahko krajše zapišemo $y = f(x)$. Množico vseh originalov imenujemo definicijsko območje funkcije f in jo označimo z D_f, množico vseh slik pa imenujemo zaloga vrednosti funkcije f in jo označimo z Z_f.

Poznamo več vrst funkcij, in sicer funkcijo realne spremenljivke, ki ima za podatke realna števila, realno funkcijo, ki ima za rezultate realna števila, ter realno funkcijo realne spremenljivke, ki ima za podatke in rezultate realna števila.

3.1 Linearna funkcija

Linearna funkcija je definirana s predpisom $f(x) = kx + n$ (eksplicitna oblika), kjer sta k in n realni števili.

V eksplicitni obliki premice število k imenujemo diferenčni količnik med spremembo vrednosti funkcije in spremembo vrednosti neodvisne spremenljivke. Prav tako določa naraščanje ali padanje linearne funkcije, vpliva tudi na naklonski kot funkcije. Graf linearne funkcije je premica. V premici k imenujemo smerni koeficient premice. Če premica poteka skozi dve dani točki $T1(x_1,y_1)$ in $T2(x_2,y_2)$, lahko smerni koeficient izračunamo po formuli $k = \frac{y_2-y_1}{x_2-x_1}$. Če je $k > 0$, funkcija (premica) narašča, če je $k = 0$, je funkcija konstantna (premica je vzporedna osi x) in če je $k < 0$, funkcija (premica) pada.



Slika 3: Linearna funkcija

(http://www2.arnes.si/~mpavle1/mp/lin_f.html)

Število n imenujemo začetna vrednost funkcije in določa strmino grafa. Premica seka ordinatno os pri $y = n$

ENAČBE PREMICE

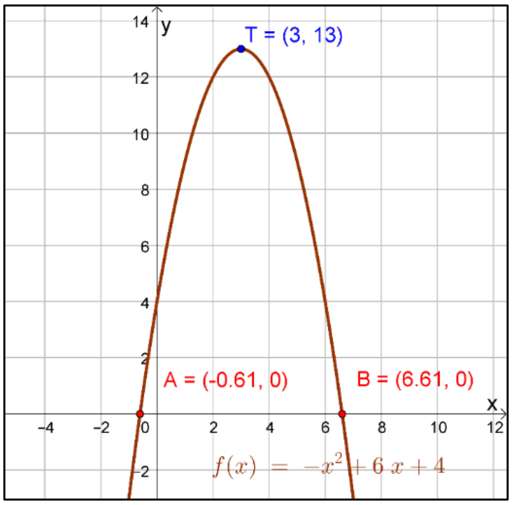
Premico lahko zapišemo v treh oblikah, in sicer v eksplicitni obliki ($y = kx + n$), implicitni obliki ($ax + by + c = 0$) ter odsekovni obliki ($\frac{x}{m} + \frac{y}{n} = 1$). Premice z enakim smernim koeficientom so vzporedne. Vzporednice imenujemo snop premic. Premice z enako začetno vrednostjo oblikujejo šop premic.

3.2 Kvadratna funkcija

Kvadratna funkcija je funkcija, ki jo lahko zapišemo v obliki $f(x) = ax^2 + bx + c$, kjer so a, b in c poljubna realna števila in je vodilni koeficient a različen od 0. Enačbo oblike $f(x) = ax^2 + bx + c$ imenujemo **splošna oblika** enačbe kvadratne funkcije.

Vsako kvadratno funkcijo lahko zapišemo tudi v **temenski obliki** $f(x) = a(x-p)^2 + q$, kjer sta števili $p\ in\ q$ koordinati temena kvadratne funkcije. Teme je točka $T(p,q)$, v kateri kvadratna funkcija doseže ekstremno vrednost. Temensko obliko lahko dobimo z dopolnjevanjem splošne oblike do popolnega kvadrata ali pa z izračunom po naslednjih formulah: $p = \frac{-b}{2a}$ in $q = \frac{-D}{4a}$. Teme lahko izračunamo tudi s pomočjo odvoda (funkcija doseže največjo oz. najmanjšo vrednost, ko je njen prvi odvod enak 0). Število D imenujemo diskriminanta ($D = b^2 - 4ac$). Le-ta vpliva posredno na teme in na ničle. Če je $D > 0$, sta obe ničli kvadratne funkcije realni, če je $D = 0$, sta števili x_1 in x_1 enaki, torej ima kvadratna funkcija samo eno, dvojno realno ničlo; če je $D < 0$, pa sta obe ničli kvadratne funkcije kompleksni, kar pomeni, da graf funkcije ne seka abscisne osi v koordinatnem sistemu.

Kvadratno funkcijo lahko zapišemo tudi v **ničelni obliki** $f(x) = a(x-x_1)(x-x_2)$, kjer sta števili x_1 in x_1 ničli kvadratne funkcije in v splošnem kompleksni števili. Ničelno obliko lahko dobimo iz splošne z razcepom. Ničli x_1 in x_1 pa lahko tudi izračunamo po formuli $x_{1,2} = \frac{-b\pm\sqrt{D}}{2a}$. Graf linearne funkcije je parabola. Narišemo ga postopoma, in sicer izračunamo presečišče parabole z ordinatno osjo in dobimo točko $Py(0,y)$, nato po formuli za p in q izračunamo teme $T(p,q)$ in na koncu izračunamo ničli kvadratne funkcije po zgoraj omenjeni formuli.



Slika 4: Kvadratna funkcija

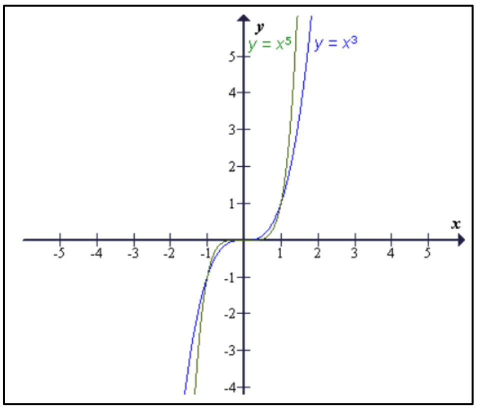
(https://edutorij.e-skole.hr/share/proxy/alfresco-noauth/edutorij/api/proxy-guest/68015b42-a876-4b53-b0e4-8723181ea03d/html/4095_Nultocke_i_ekstremi_kvadratne_funkcije.html)

3.3 Potenčna funkcija

Potenčna funkcija je funkcija, ki jo lahko zapišemo z enačbo oblike $f(x) = x^n$, kjer osnova x zavzame poljubno realno število, eksponent pa je izbrano celo število. Potenčno funkcijo razdelimo v štiri skupine, opisane v nadaljevanju.

3.3.1 Potenčne funkcije z lihim pozitivnim eksponentom

Funkcije iz te skupine imajo določene skupne lastnosti. Za njih je značilno, da imajo število v eksponentu večje od 1. Grafi se po obliki razlikujejo od simetrale lihih kvadrantov $y = x$, saj je slednja premica. Te funkcije lihe stopnje so naraščajoče in lihe na celotnem definicijskem območju. Definicijsko območje in zaloga vrednosti so vsa realna števila. Njihovi grafi so simetrični glede na koordinatno izhodišče.

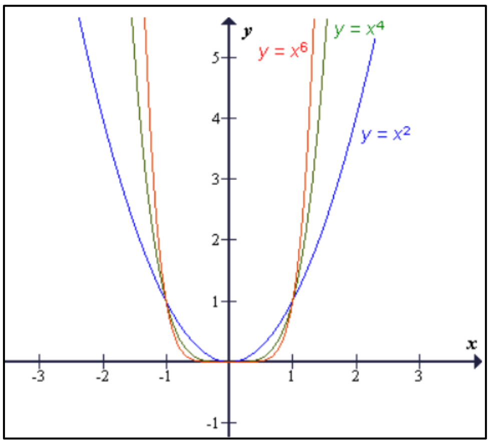


Slika 5: Potenčne funkcije z lihim pozitivnim eksponentom

(http://www2.arnes.si/~mpavle1/mp/pot_f.html)

3.3.2 Potenčne funkcije s sodim pozitivnim eksponentom

Funkcije iz te skupine imajo določene skupne lastnosti. Funkcije naraščajo, kjer so $x > 0$ in padajo, kjer so $x < 0$. Njihovi grafi (Slika 6) so simetrični glede na ordinatno os, kar pomeni, da so funkcije sode. Definicijsko območje so vsa realna števila. Zaloga funkcije so vsa nenegativna števila.

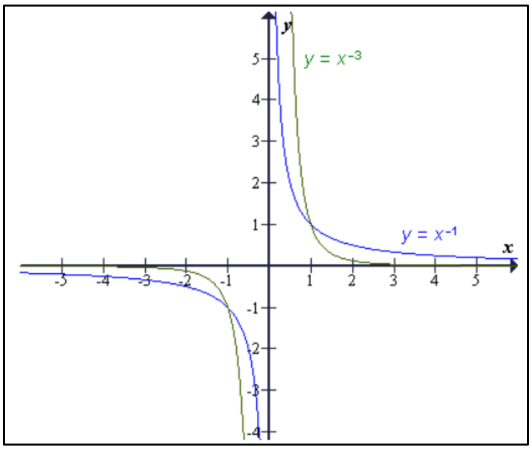


Slika 6: Potenčne funkcije s sodim pozitivnim eksponentom

(http://www2.arnes.si/~mpavle1/mp/pot_f.html)

3.3.3 Potenčne funkcije z lihim negativnim eksponentom

Skupne lastnosti funkcij, ki sodijo v to skupino potenčnih funkcij, so, da grafi potekajo skozi točko M(1,1) in N(-1,-1) (Slika 7). Dane funkcije so neomejene, vendar niso definirane pri $x = 0$. Funkcije so lihe, saj so simetrične glede na koordinatno izhodišče. Imajo vodoravno asimptoto $y = 0$ ter navpično asimptoto (pol) $x = 0$. Zaradi omenjenih asimptot dani grafi nimajo ničel ter se daleč od izhodišča (proti neskončnosti) približujejo vrednosti funkcije 0. Graf se asimptotsko približuje obema osema, vendar se ju nikoli ne dotakne ali preseka. Grafi prav tako padajo na intervalu $(-\infty, 0)$ in na intervalu $(0, \infty)$.

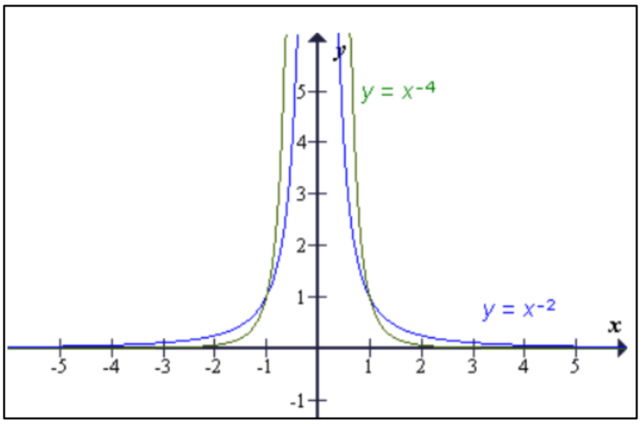


Slika 7: Potenčne funkcije z lihim negativnim eksponentom

(http://www2.arnes.si/~mpavle1/mp/pot_f.html)

3.3.4 Potenčne funkcije s sodim negativnim eksponentom

Lastnosti, ki so skupne funkcijam iz te skupine potenčnih funkcij, se odražajo pri grafih, saj je le-ta simetričen glede na ordinatno os, torej so to sode funkcije (Slika 8). Prav tako so grafi vedno narisani nad abscisno osjo. Funkcije naraščajo na intervalu (-∞,0) in padajo na intervalu $(0, \infty)$. Funkcije imajo navpično asimptoto $x = 0$ in vodoravno asimptoto $y = 0$, zato se grafi asimptotsko približujejo obema osema.



Slika 8: Potenčne funkcije s sodim negativnim eksponentom

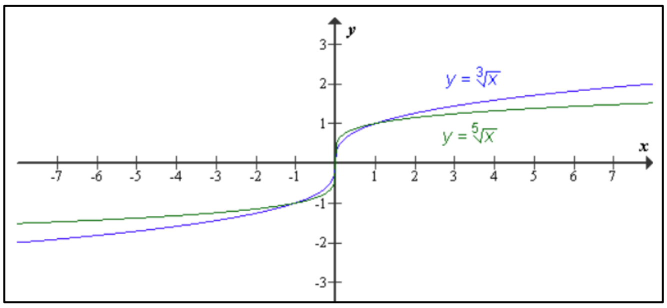
(http://www2.arnes.si/~mpavle1/mp/pot_f.html)

3.4 Korenska funkcija

Korenska funkcija je vsaka funkcija, ki jo lahko zapišemo z enačbo oblike $f(x) = \sqrt[n]{x}$, kjer je število n naravno število in večje od 1. Prav tako pri korenski funkciji ločimo dve vrsti.

3.4.1 Korenske funkcije z lihim korenskim eksponentom

Funkcije iz te skupine naraščajo, so neomejene in so lihe funkcije. V okolici koordinatnega izhodišča je graf funkcije navpičen (Slika 9). Funkcije imajo ničle pri x=0. Pri $x < 0$ so funkcije negativne, pri $x > 0$ pa je pozitivne. Definicijsko območje so vsa realna števila.

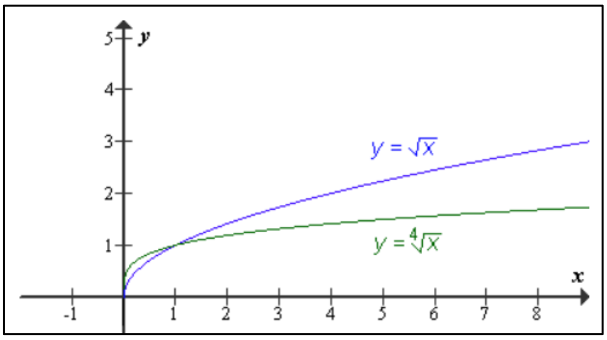


Slika 9: Korenske funkcije z lihim korenskim eksponentom

(http://www2.arnes.si/~mpavle1/mp/kor_f.html)

3.4.2 Korenske funkcije s sodim korenskim eksponentom

Funkcije iz te skupine naraščajo, so omejene in niso ne lihe in ne sode (Slika 10). Definicijsko območje in zaloga vrednosti funkcije so vsa nenegativna realna števila.



Slika 10: Korenske funkcije s sodim korenskim eksponentom

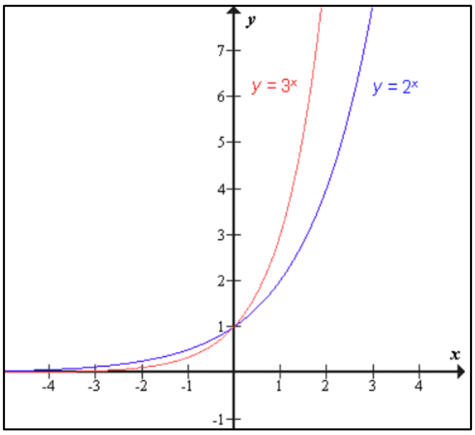
(http://www2.arnes.si/~mpavle1/mp/kor_f.html)

3.5 Eksponentna funkcija

Eksponentna funkcija je funkcija, katere enačbo lahko zapišemo v obliki $f(x) = a^x$, kjer mora biti število a pozitivno, vendar ne sme biti 1. V primeru, če bi bil a enak 1, ne dobimo »pravo« eksponentno funkcijo, saj velja, da je $f(x) = 1^x = 1$ za poljubno realno število x. Za to funkcijo je značilno, da ob večanju neodvisne spremenljivke, funkcijske vrednosti rastejo eksponentno. Kot vse omenjene funkcije, tudi to razdelimo na dve skupini. Skupini ločimo glede na osnovo a.

3.5.1 Eksponentna funkcija z osnovo a > 1

Funkcije so definirane za vsa realna števila, zaloga vrednosti je množica pozitivnih realnih števil. Grafi sekajo ordinatno os v točki M(0,1) (Slika 11). Funkcije so naraščujoče in navzdol omejene z 0, navzgor pa niso omejene. Imajo vodoravno asimptoto $y = 0$, torej se grafi asimptotsko približujejo abscisni osi.

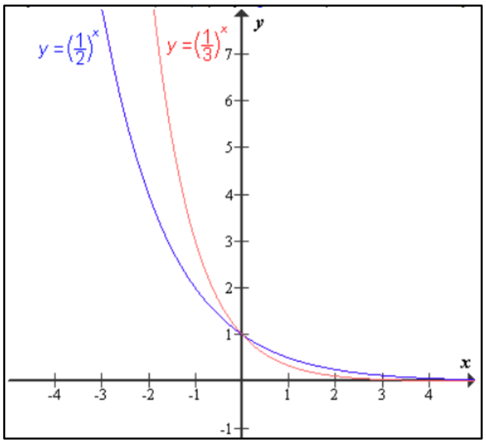


Slika 11: Osnova večja od 1

(http://www2.arnes.si/~mpavle1/mp/exp_f.html)

3.5.2 Eksponentna funkcija z osnovo 0 < a < 1

Funkcije so definirane za vsa realna števila in zaloga vrednosti je množica pozitivnih realnih števil. Grafi prav tako sekajo ordinatno os v točki M(0,1) (Slika 12). Funkcije so padajoče, navzdol omejene z 0, navzgor pa niso omejene. Imajo vodoravno asimptoto $y = 0$, torej se grafi asimptotsko približujejo abscisni osi.



Slika 12: Osnova a manjša od 1 in večja od 0

(http://www2.arnes.si/~mpavle1/mp/exp_f.html)

3.6 Logaritemska funkcija

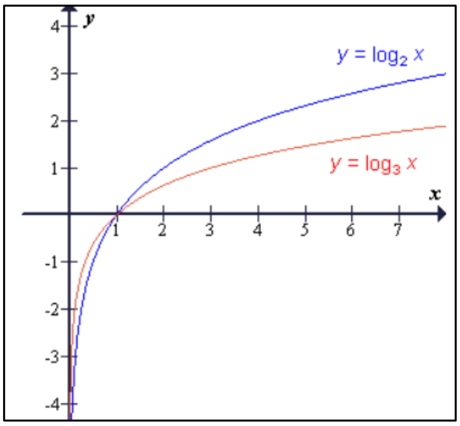
Preden opišem logaritemsko funkcijo, bom opisala, kaj je logaritem.

Logaritem $\log_a x$ je sestavljen iz logaritmanda (x) in osnove logaritma (a). Najbolj znana sta logaritem z osnovo 10 in naravni logaritem z osnovo Eulerjevo število e (e = 2.71828…). Označimo ju kot $log_{10}b = logb\ in\ \log_e b = \ln b$.

Logaritemska funkcija je inverzna funkcija eksponentne funkcije. Torej je logaritem števila b pri osnovi a tisti eksponent x, za katerega velja $a^x = b$. Torej če matematično zapišem: $\log_a b = x \Leftrightarrow a^x = b$. Logaritemska funkcija mora imeti tako kot eksponentna funkcija osnovo pozitivno in različno od 1, zato se tudi logaritemske funkcije delijo v dve skupini:

3.6.1 Logaritemska funkcija z osnovo a > 1

Logaritemske funkcije v tej skupini naraščajo povsod, kjer je definirana (Slika 13) in ima ničlo pri $x = 1$. Funkcije so definirane za vsa pozitivna realna števila in zaloga vrednosti je množica realnih števil. Asimptota je ordinatna os. Funkcije so prav tako navzdol in navzgor neomejene.

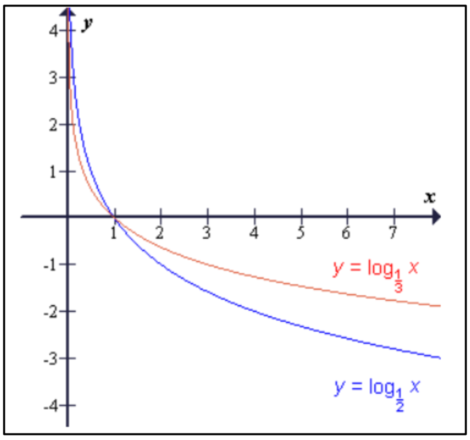


Slika 13: Logaritemska funkcija z osnovo a>1

(http://www2.arnes.si/~mpavle1/mp/log_f.html)

3.6.2 Logaritemska funkcija z osnovo 0 < a < 1

Funkcije so prav tako definirane za vsa pozitivna realna števila, zaloga vrednosti je množica realnih števil. Že iz grafa funkcij se vidi, da funkcije povsod na definicijskem območju padajo. Funkcije imajo ničlo pri $x = 1$ in navpično asimptoto $x = 0$. Funkcije so prav tako navzdol in navzgor neomejene.



Slika 14: Logaritemska funkcija z sonovo 0<a<1

(http://www2.arnes.si/~mpavle1/mp/log_f.html)

3.7 Polinomska funkcija

3.7.1 Polinomi

Polinomi so realne funkcije realne spremenljivke, ki so definirane za vsa realna števila. Oblika zapisa je $p(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_2 x^2 + a_1 x + a_0$. Števila $\boldsymbol{a_0}, \boldsymbol{a_1}, \boldsymbol{a_2}, \boldsymbol{a_3} \ldots \boldsymbol{a_n}$ so koeficienti polinoma $p(x)$, $\boldsymbol{a_n}$ je vodilni koeficient (po dogovoru ne sme biti 0), $\boldsymbol{a_0}$ je prosti člen, naravno število $\boldsymbol{n}$ pa je stopnja polinoma $p(x)$.

3.7.2 Računanje s polinomi

<u>SEŠTEVANJE POLINOMOV</u>

Dva polinoma $p(x)$ in $q(x)$ seštejemo tako, da seštejemo koeficiente pri potencah z istim eksponentom. Vsota je spet polinom. Njegova stopnja je enaka višji od stopenj obeh polinomov, ki ju seštevamo. Stopnja vsote se zniža, če sta seštevanja iste stopnje in imata nasprotna vodilna koeficienta. Seštevamo lahko več polinomov na enak način.

<u>ODŠTEVANJE POLINOMOV</u>

Od polinoma $p(x)$ odštejemo $q(x)$ tako, da odštejemo koeficiente $q(x)$ od istoležnih koeficientov $p(x)$. Polinom, ki je enak razliki polinomov, ima v splošnem stopnjo enako višji od stopenj obeh polinomov. Samo v primeru, če odštevamo polinoma enake stopnje z enakim vodilnima koeficientoma, ima razlika polinomov nižjo stopnjo.

<u>MNOŽENJE POLINOMOV</u>

Polinoma $p(x)$ in $q(x)$ pomnožimo tako, da pomnožimo vsak člen polinoma $p(x)$ z vsakim členom polinoma $q(x)$. Produkt polinomov je polinom, čigar stopnja je enaka vsoti stopenj faktorjev. Množimo lahko več polinomov na enak način.

<u>DELJENJE POLINOMOV</u>

Polinom $p(x)$ lahko tudi delimo s $q(x)$. Pogoj in postopek deljenja temelji na Osnovnem izreku o deljenju naravnih števil. Posledično pomeni, da je stopnja delitelja $q(x)$ enaka ali manjša od stopnje polinoma $p(x)$.

Deljenje polinoma $p(x)$ in $q(x)$ lahko zapišemo kot:

$$p(x) = k(x) * q(x) + o(x),$$

kjer je $q(x)$ delitelj, $p(x)$ deljenec, $k(x)$ količnik in $o(x)$ ostanek. Če je $o(x)$ enak nič, pomeni, da sta polinoma $p(x)$ in $q(x)$ deljiva, oziroma da $q(x)$ deli $p(x)$.

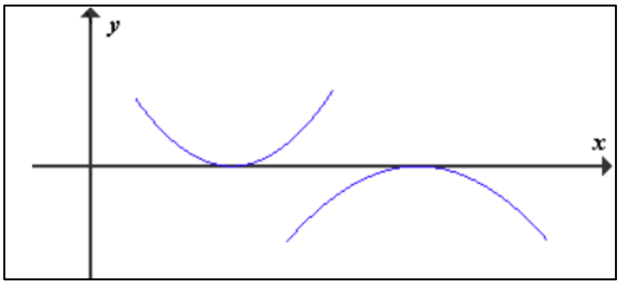
3.7.3 Vpliv ničel na graf polinoma

Polinom je zvezna funkcija. To pomeni, da je graf polinoma nepretrgana krivulja in da vsebuje neskončno točk. Kot pri vsakem grafu funkcije, je tudi tu pomembno njeno vedenje daleč od izhodišča (ko x limitira proti $\pm\infty$). Daleč od izhodišča vrednosti funkcije tudi limitirajo proti $\pm\infty$. Pri risanju grafa imajo za srednješolsko izobraževanje od vseh točk najpomembnejšo vlogo presečišča z osema.

Če za polinom $p(x_1)$ velja, da je njegova vrednost 0, je x_1 ničla polinoma. Ničle so abscise presečišč grafa polinoma z abscisno osjo.

Poznamo ničle sode in ničle lihe stopnje oz. sode in lihe ničle. Te ničle različno vplivajo na potek grafa polinoma. Liho oz. sodo stopnjo določa število enakih ničel in ne njihova vrednost (npr. x=2 je ničla prve stopnje, torej lihe stopnje, $x_{1,2}$ = 3 je ničla druge stopnje, torej sode stopnje). Ničle vplivajo tudi na naklonski kot grafa polinoma.

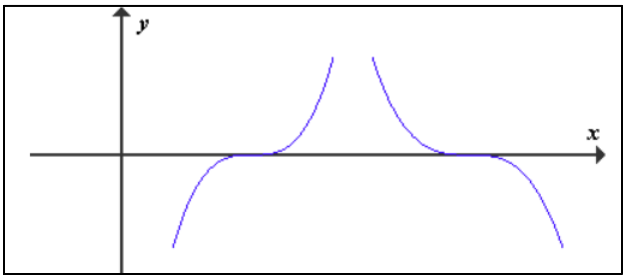
SODE NIČLE – pri prehodu grafa skozi sodo ničlo, se graf samo dotakne osi x. Posledično funkcija ne spremeni predznaka. V ničli sode stopnje lahko ima polinom lokalni ekstrem (minimum ali maksimum). Pri prehodu skozi te ničle se spremeni tudi lastnost naraščanja oz. padanja.



Slika 15: Ničle sode stopnje

(http://www2.arnes.si/~mpavle1/mp/polinom.html)

LIHE NIČLE – pri prehodu grafa skozi liho ničlo, graf seka os x. Posledično funkcija spremeni predznak. Pri prehodu skozi te ničle se ne spremeni lastnost naraščanja oz. padanja.



Slika 16: Ničle lihe stopnje, ki so večje od 1

(http://www2.arnes.si/~mpavle1/mp/polinom.html)

3.8 Racionalna funkcija

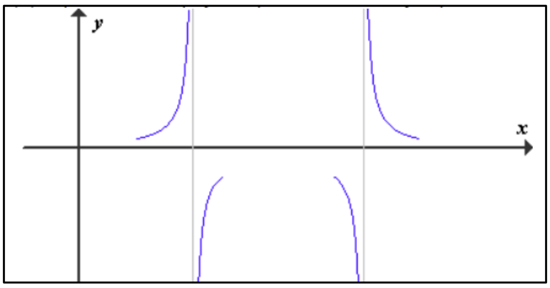
Racionalna funkcija je vsaka funkcija oblike $f(x) = \frac{p(x)}{q(x)}$, kjer je $p(x)$ poljuben polinom, $q(x)$ pa poljuben neničelni polinom. Pogoj za obstoj te funkcije je, da je zapisan ulomek okrajšan. To pomeni, da je največji skupni delitelj $p(x)$ in $q(x)$ ena. Ker je funkcija odvisna od polinomov, jo lahko zapišemo tudi kot $f(x) = \frac{a_n x^n + a_{n-1} x^{n-1} + \cdots + a_2 x^2 + a_1 x + a_0}{b_m x^m + b_{m-1} x^{m-1} + \cdots + b_2 x^2 + b_1 x + b_0}$.

Racionalna funkcija je zvezna povsod, kjer je definirana. To pomeni, da ni njen graf neprekinjena krivulja. Kot pri vsakem grafu funkcije, je tudi pri tej pomembno njeno vedenje daleč od izhodišča (ko x limitira proti $\pm\infty$). Daleč od izhodišča se graf funkcije približuje asimptoti. Ta asimptota je količnik, ki ga dobimo, če $p(x)$ delimo s $q(x)$.

Pri risanju grafa imajo za srednješolsko izobraževanje od vseh točk najpomembnejšo vlogo presečišča z osema. Pomen in vrste ničel ter njihov vpliv je enak kot pri polinomih, saj je racionalna funkcija sestavljena iz polinomov.

Pomembno vlogo pa imajo tudi ničle v imenovalcu. Te ničle predstavljajo pole oz. navpične asimptote. Zaradi omenjene definicije polov ločimo tudi pole (tako kot ničle) na sode in lihe stopnje.

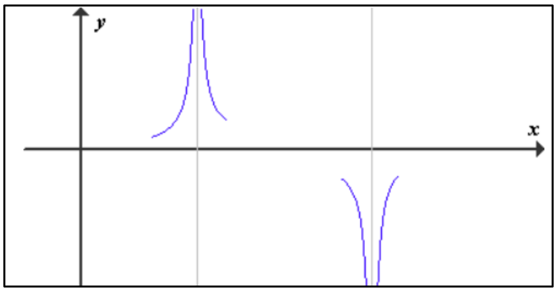
LIHI POLI – pri prehodu grafa skozi lihi pol funkcija spremeni predznaka. Ne spremeni se pa njena lastnost naraščanja oz. padanja.



Slika 17: Poli lihe stopnje

(http://www2.arnes.si/~mpavle1/mp/rac_f.html)

SODI POLI – pri prehodu grafa skozi sodi pol funkcija ne spremeni predznaka. Spremeni pa se njena lastnost naraščanja oz. padanja.



Slika 18: Poli sode stopnje

(http://www2.arnes.si/~mpavle1/mp/rac_f.html)

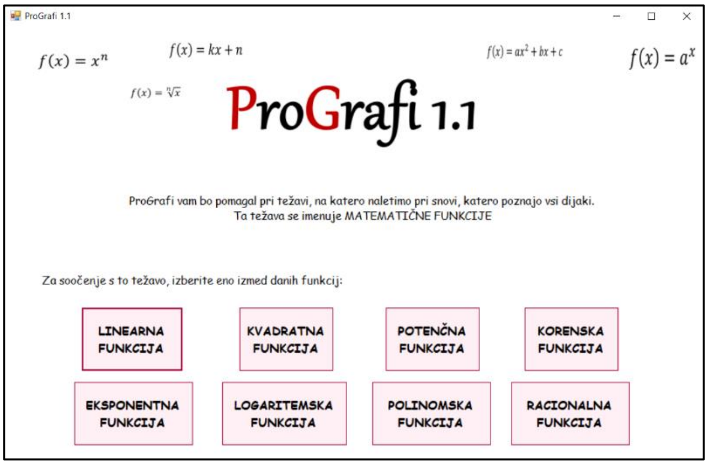
4 APLIKACIJA PROGRAFI 1.1

4.1 Namen in nadgradnja aplikacije

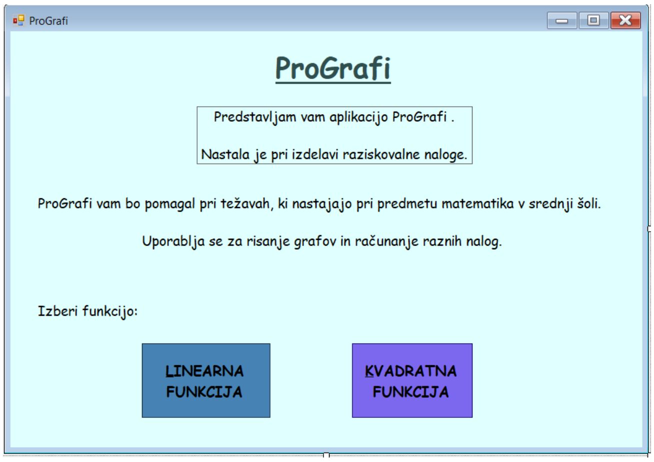
V lanskem šolskem letu sem izdelala aplikacijo Prografi, katere glavni namen je dijakom olajšati učenje matematičnih funkcij. Ker je zavzemala samo dve matematični funkciji (linearno in kvadratno), sem jo želela nadgraditi z večjo izbiro funkcij. Dodala sem več različnih tipskih nalog ter nekaterim funkcijam še dodatno razlago o njihovih lastnostih. Tako je nastala novejša verzija, ki se imenuje ProGrafi 1.1. Namen aplikacije ostaja enak, vendar sem želela podrobneje implementirati še ostale funkcije ter nadgraditi obstoječo.

Ker z izgledom prejšnje verzije nisem bila zadovoljna, sem ga želela posodobiti. Na Sliki 20 vidimo, kako je aplikacija izgledala v prejšnji verziji, poleg nje pa je na Sliki 19 izgled novejše verzije. Nadgradnja te aplikacije vsebuje tudi svoj logotip, ki ga vidimo na začetni strani.

V nadaljevanju bom opisala celotno aplikacijo, njen izgled, kako deluje, nato pa bom opisala kodo in obrazložila, kako aplikacija računa ter prikazuje rešitve različnih tipskih nalog in kako se funkcije rišejo.



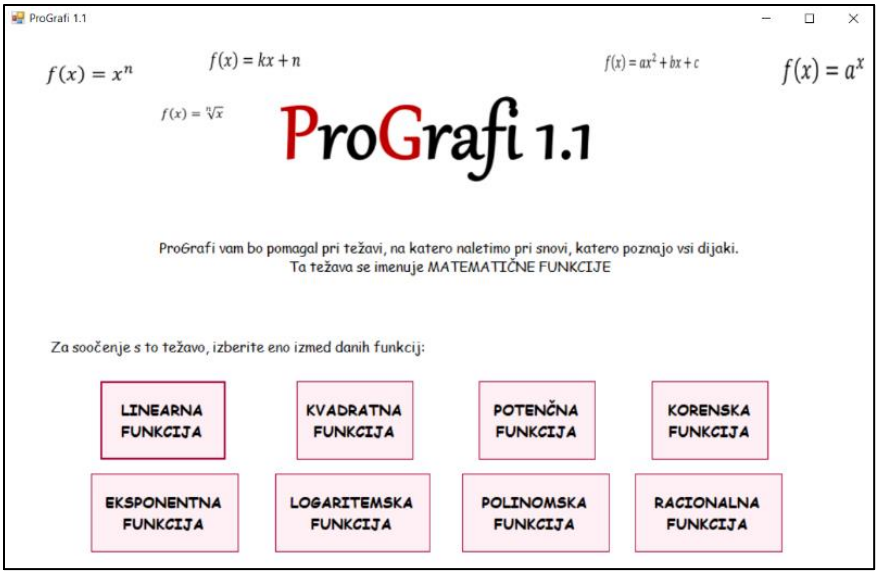
Slika 19: Novejša verzija - ProGrafi 1.1



Slika 20: Starejša verzija - ProGrafi

4.2 Delovanje aplikacije

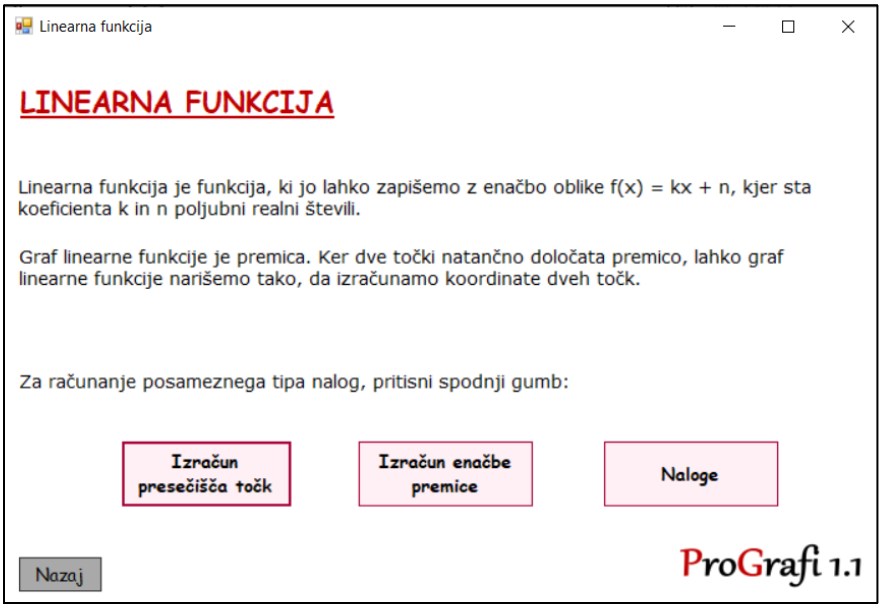
Ob zagonu programa se najprej prikaže začetna stran, ki vsebuje kratek opis ter vse matematične funkcije, ki jih zajema. Funkcije si sledijo na podlagi obravnavane snovi pri predmetu matematika v srednjih šolah. Začetno stran lahko vidimo na Sliki 21.



Slika 21: Začetna stran aplikacije

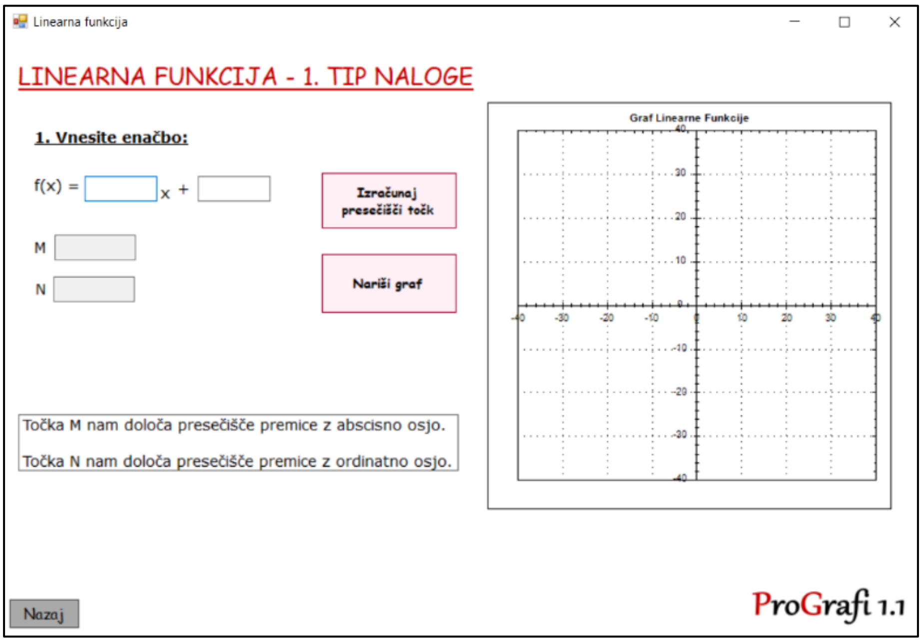
4.2.1 Linearna funkcija

Pri izbiri gumba za linearno funkcijo se prikaže osnovna stran (Slika 22), kjer lahko preberemo kratko razlago o tem, kaj je linearna funkcija. Po razlagi vidimo tri gumbe. Vsak gumb vsebuje različni tip naloge. Klik na enega od gumbov sproži t. i. dogodek in odpre novo okno, trenutnega pa zapre.



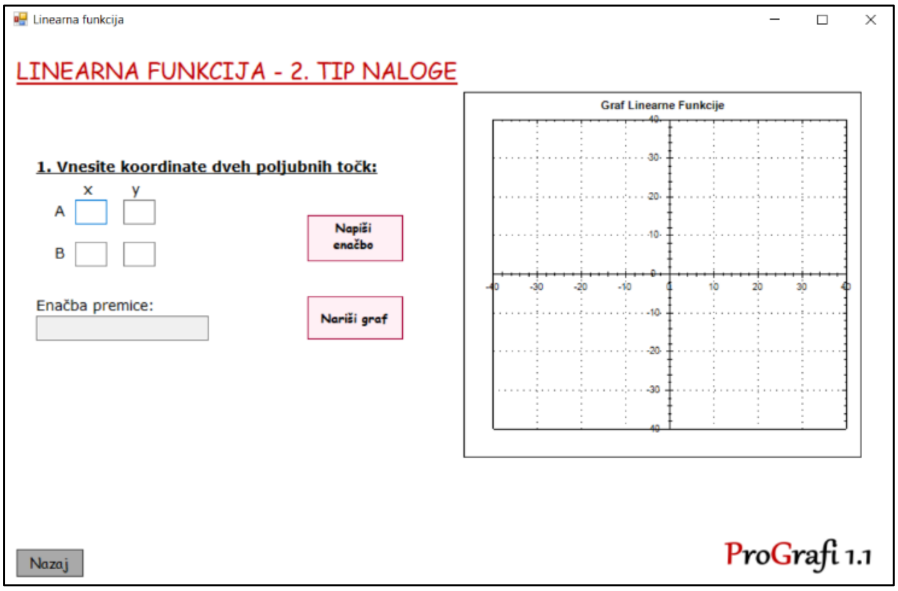
Slika 22: Začetna stran Linearna funkcije

Pri gumbu »Izračun presečišča točk« se pojavi prvi tip naloge (Slika 23). Najprej vpišemo enačbo premice, program pa izračuna presečišče "N" z ordinatno osjo ter presečišče "M" z abscisno osjo. S pritiskom na gumb »Nariši graf« program nariše graf glede na vnesene podatke.



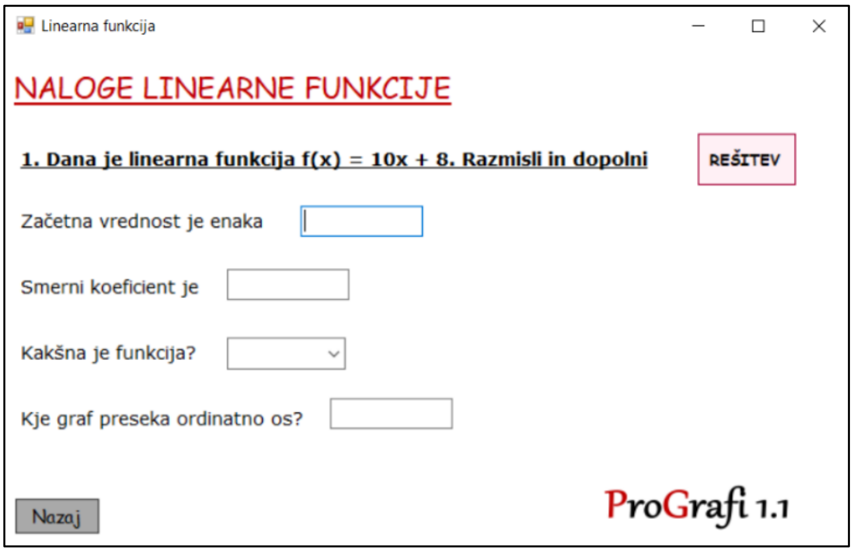
Slika 23: Prvi tip naloge linearne funkcije

Pri gumbu »Izračun enačbe premice« se prikaže drugi tip naloge (Slika 24). Uporabnik vpiše koordinate dveh točk in program izračuna enačbo premice, ki poteka skozi njiju. Ta dva tipa nalog je vključevala že prejšnja verzija.



Slika 24: Drugi tip naloge linearne funkcije

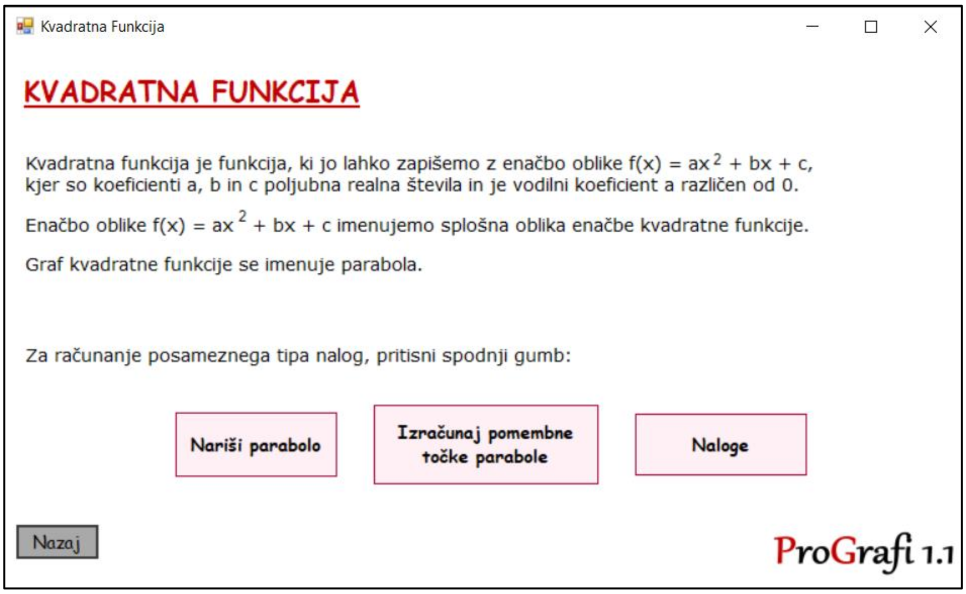
Linearni funkciji sem tako dodala še tretji gumb »Naloge«, ki odpre novo okno z novo dodanim tipom naloge (Slika 25). Pri tem tipu naloge aplikacija poda funkcijo, uporabnik pa mora sam vpisati začetno vrednost, smerni koeficient in presečišče grafa z ordinatno osjo. Prav tako mora določiti, ali je funkcija naraščajoča ali padajoča. Poleg opisanega tipa naloga ima uporabnik zraven še gumb »Rešitev«, da lahko preveri, če je odgovoril pravilno.



Slika 25: Naloge linearne fukcije

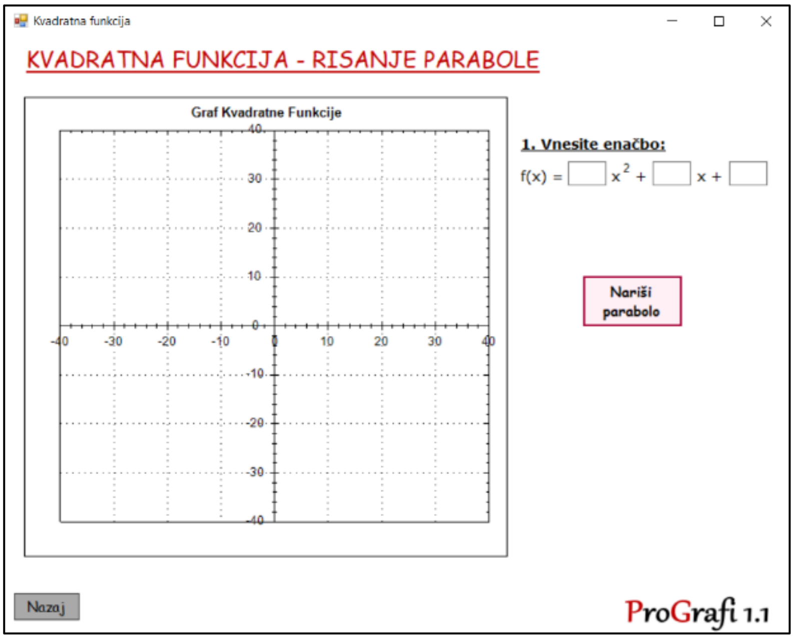
4.2.2 Kvadratna funkcija

Na začetku je razložena definicija kvadratne funkcije (Slika 26). Za zapisano definicijo sledijo trije gumbi, ki vsebujejo tri različne tipe nalog.



Slika 26: Začetna stran kvadratne funkcije

Prvi tip naloge vsebuje novo okno za risanje grafa kvadratne funkcije oziroma parabole, ki ga lahko vidimo na Sliki 28. Uporabnik mora vpisati dano enačbo ter pritisniti gumb »Nariši parabolo«. S klikom na gumb aplikacija nariše potrebno enačbo.



Slika 27: Prvi tip naloge kvadratne funkcije

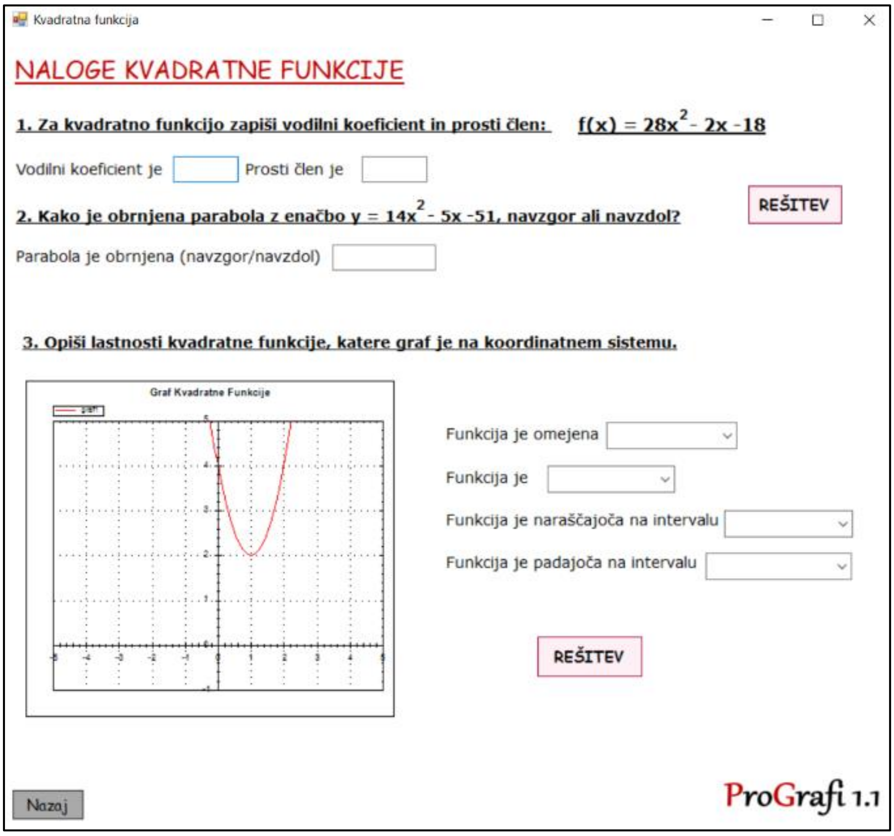
Pri drugem tipu naloge nam aplikacija glede na vneseno enačbo izračuna pomembne točke parabole, to so teme, presečišče ordinatne osi in ničli funkcije (Slika 28). Podrobnejši opis teh pomembnih točk je v poglavju 3.2 (Kvadratne funkcije). Zraven je prikazan tudi okvirček s kratko razlago o izračunanih pomembnih točkah.

Slika 28: Drugi tip naloge kvadratne funkcije

Novejša verzija ProGrafi 1.1 vsebuje še tretji tip naloge, in sicer različne tipske naloge (Slika 29). S klikom na gumb »Naloge« se odpre novo okno, kjer so prikazane tri različne naloge. Pri prvi nalogi ima uporabnik podano funkcijo, v kateri mora dopolniti vodilni koeficient in prosti člen.

Druga naloga od uporabnika zahteva določitev napisane funkcije, če je parabola obrnjena navzgor ali navzdol.

Pri tretji nalogi ima uporabnik narisano parabolo in mora iz nje razbrati, če je omejena, njeno sodost/lihost, na katerem intervalu je naraščajoča ter na katerem padajoča. Poleg opisanih nalog ima uporabnik zraven še dva gumba »Rešitev«, da lahko preveri, če je odgovoril pravilno.



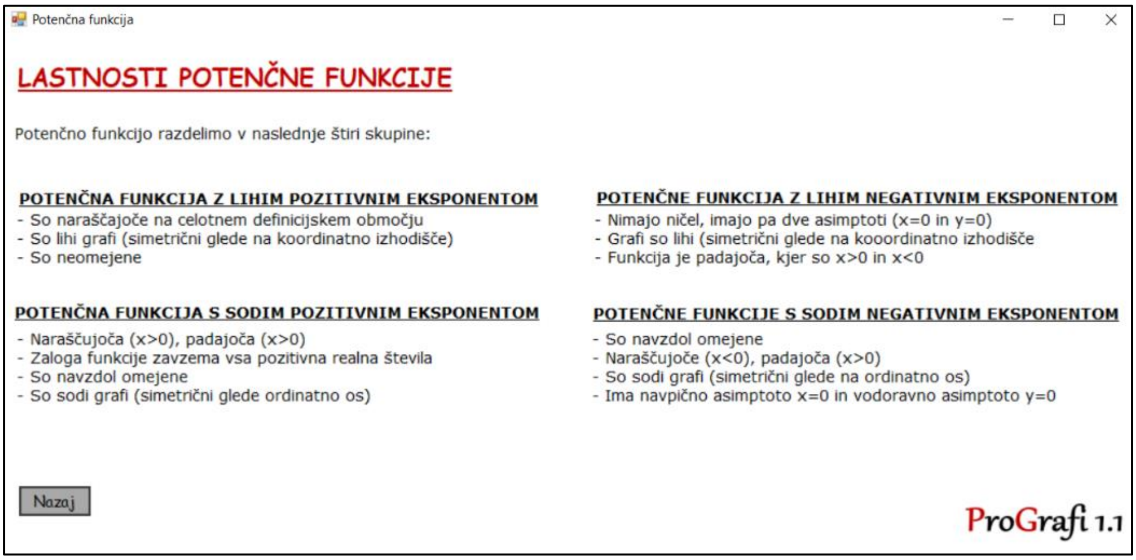
Slika 29: Naloge kvadratne funkcije

4.2.3 Potenčna funkcija

Na začetni strani potenčne funkcije je kratka razlaga, kaj potenčna funkcija je (Slika 30). Na dnu so prikazani trije gumbi.

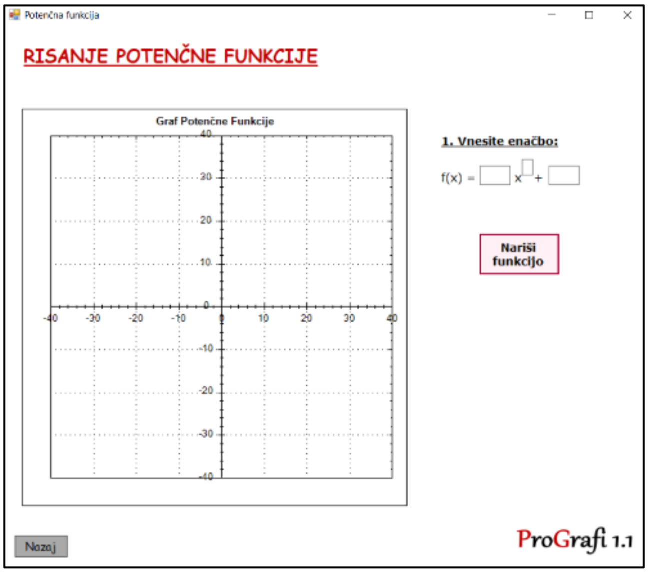
Slika 30: Začetna stran potenčne funkcije

Pri kliku na gumb »Lastnosti funkcije« se prikaže novo okno, kjer so opisane vse štiri skupine, v katere razdelimo potenčno funkcijo (Slika 31). Vsebuje razlago o potenčni funkciji z naravnim pozitivnim in negativnim eksponentom.



Slika 31: Lastnosti potenčne funkcije

Gumb »Risanje funkcije« omogoča uporabniku, da lahko preveri, če je pri domači nalogi v šoli pravilno narisal potenčno funkcijo (Slika 32). Aplikacija na vpisane podatke sama nariše pravilno funkcijo.

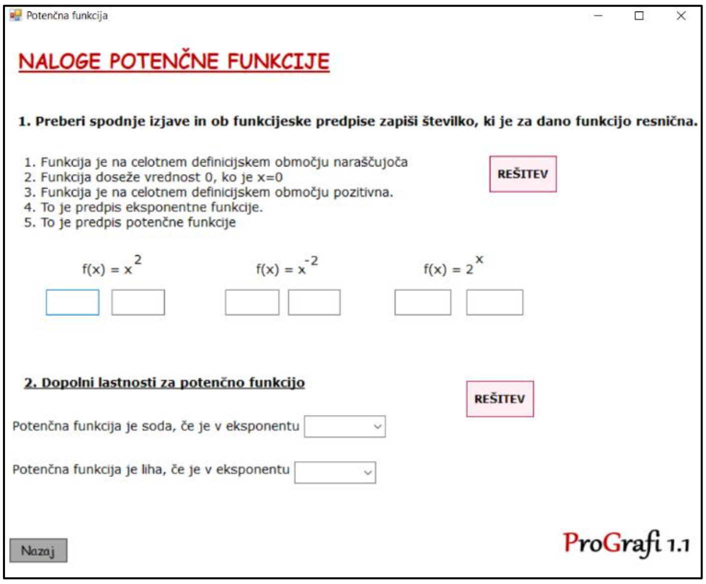


Slika 32: Risanje potenčne funkcije

Svoje znanje o teoriji potenčne funkcije pa uporabnik preveri ko klikne na gumb »Teoretične naloge«. Prikažeta se dve različni nalogi (Slika 33).

Prva naloga vsebuje 5 izjav in tri različne zapise funkcij. Uporabnik mora tako dopisati k vsaki funkciji številko izjave, ki tej funkciji pripada.

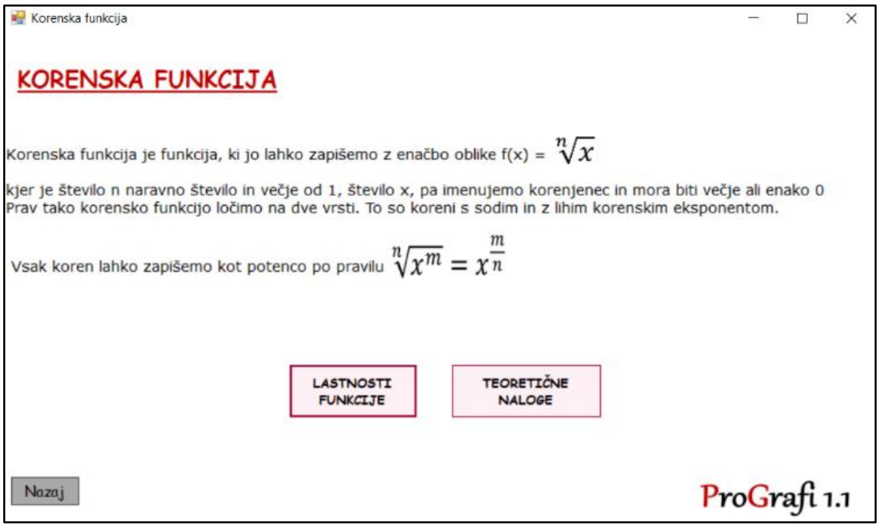
Pri drugi nalogi mora uporabnik dopolniti lastnosti, ki veljajo za potenčno funkcijo. Zraven obeh nalog se nahaja tudi gumb »Rešitev«, kamor uporabnik klikne, ko želi preveriti, če je odgovoril pravilno.



Slika 33: Naloge potenčne funkcije

4.2.4 Korenska funkcija

Na začetni strani je najprej definicija korenske funkcije, sledita pa ji dva gumba, kjer prvi predstavlja lastnosti funkcije, drugi pa teoretične naloge, ki jih uporabnik rešuje sam (Slika 34).

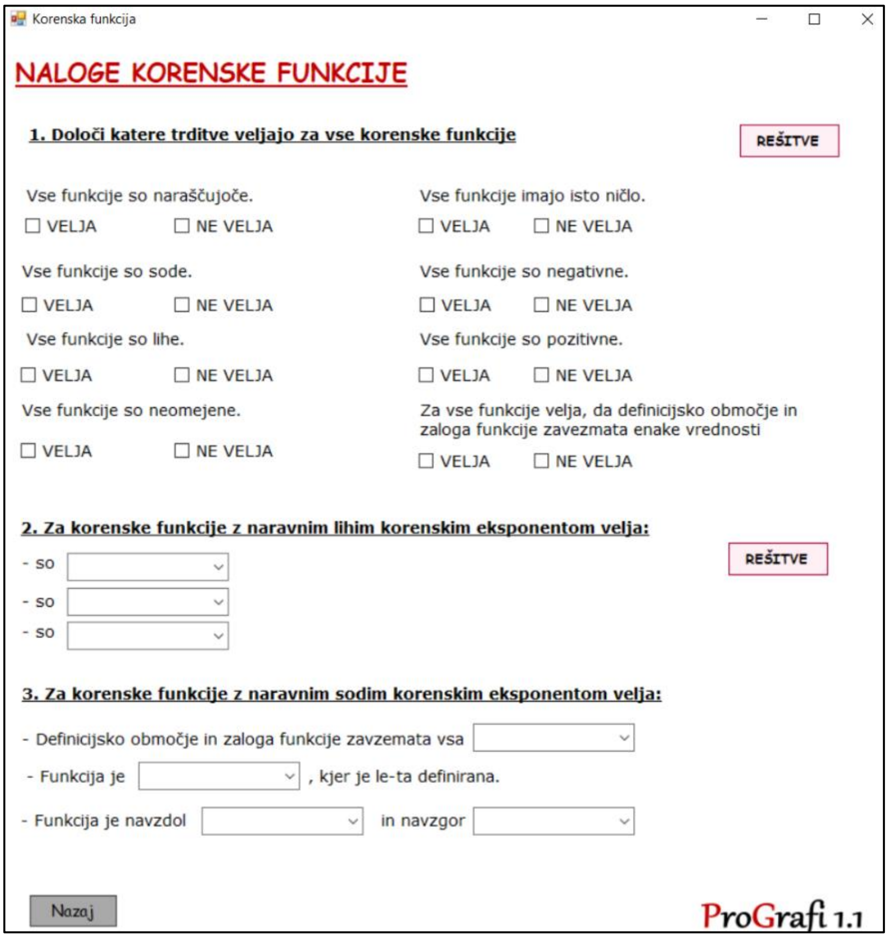


Slika 34: Začetna stran korenske funkcije

Klik na gumb »Lastnosti funkcije« nas popelje na novo okence, kjer se pojavijo lastnosti, kot pove že samo ime gumba (Slika 35). Podrobneje sta opisani predvsem dve skupini, v kateri delimo korensko funkcijo – torej korenska funkcija z lihim in sodim korenskim eksponentom.

Slika 35: Lastnosti korenske funkcije

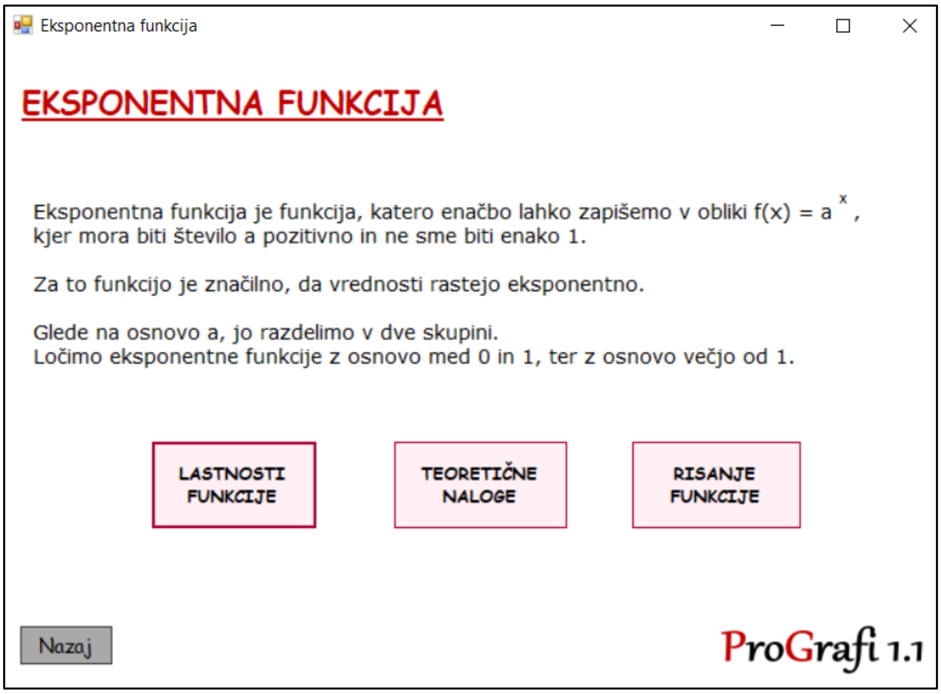
Gumb »Teoretične naloge« pomagajo uporabniku ponoviti teorijo in lastnosti, ki jih je spoznal preko treh različnih tipskih nalog (Slika 36). Pri prvi nalogi mora uporabnik označiti, ali zapisana trditev drži ali ne drži za vse korenske funkcije. Za tem mora določiti, kaj velja za korenske funkcije z naravnim lihim korenskim eksponentom in kaj velja za korenske funkcije z naravnim sodim korenskim eksponentom. Pri vsaki nalog je gumb »Rešitve«, kjer lahko uporabnik preveri dane odgovore, ki jih je izbral.



Slika 36: Naloge korenske funkcije

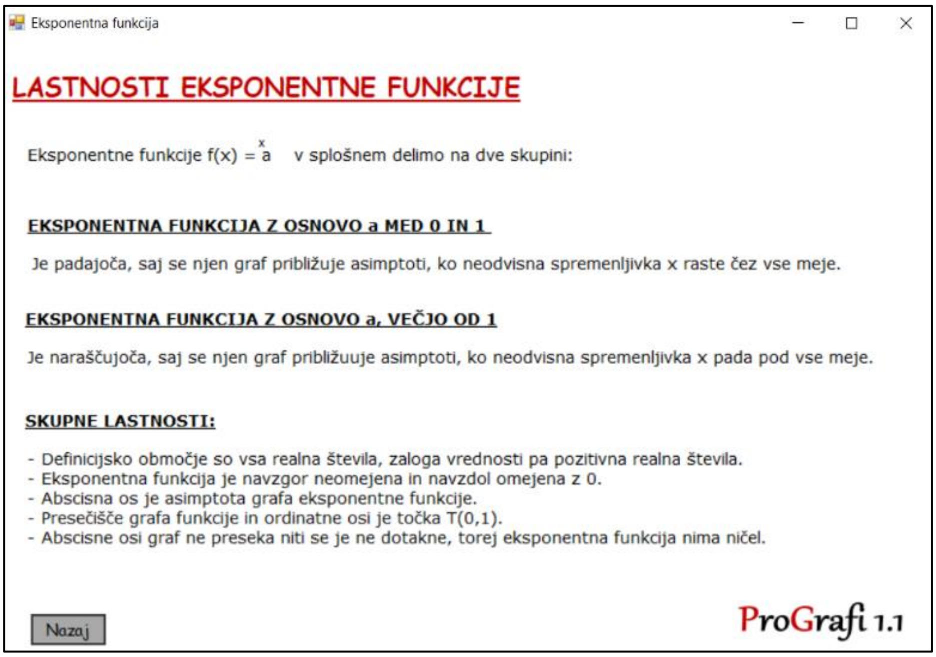
4.2.5 Eksponentna funkcija

Eksponentna funkcija vsebuje na začetni strani kratko teorijo, sledijo pa ji trije različni gumbi, ki vsebujejo različne tipe nalog (Slika 37).



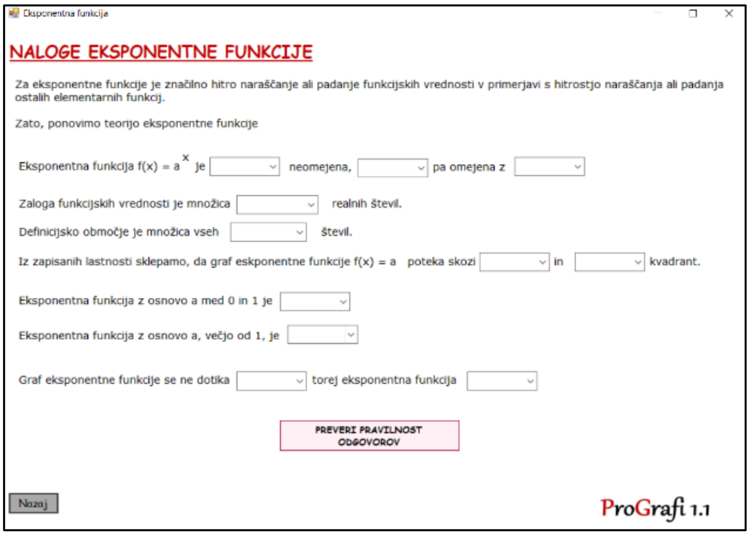
Slika 37: Začetna stran eksponentne funkcije

Prvi gumb »Lastnosti funkcije« vsebuje razlago o lastnostih, ki jih mora poznati vsak dijak (Slika 38). Pove nam, kakšna je funkcija, kadar ima osnovo med 0 in 1 ter kadar ima osnovo večjo od 1. Prav tako so opisane še nekatere lastnosti, ki so skupne, ne glede na to kakšno osnovo ima funkcija.



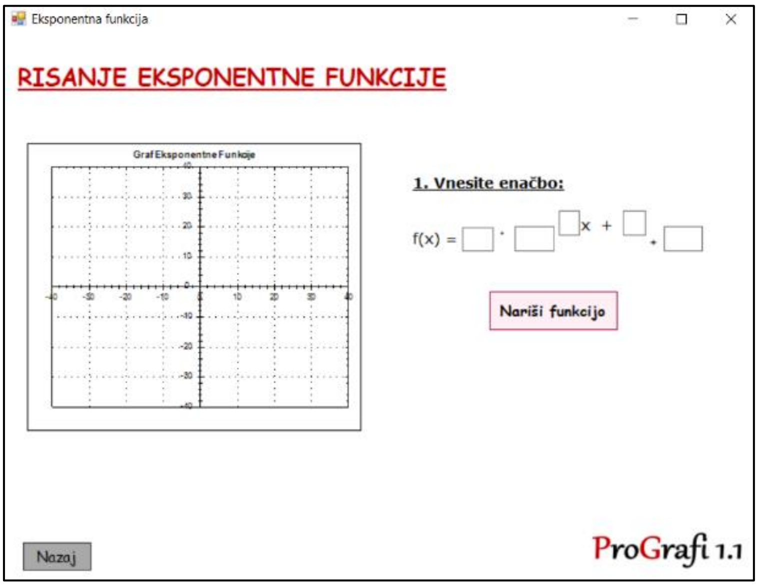
Slika 38: Lastnosti eksponentne funkcije

Drugi gumb »Teoretične naloge« vsebuje ponovitvene naloge o teoriji, ki jo je uporabnik spoznal preko aplikacije ali v šoli (Slika 39). Za vsako poved, ki je prikazana, mora uporabnik izbrati med možnostmi, katera beseda oziroma izbira je pravilna. Če uporabnik klikne na gumb (Preveri pravilnost odgovorov), bo izvedel, ali je odgovoril pravilno ali ne.



Slika 39: Naloge eksponentne funkcije

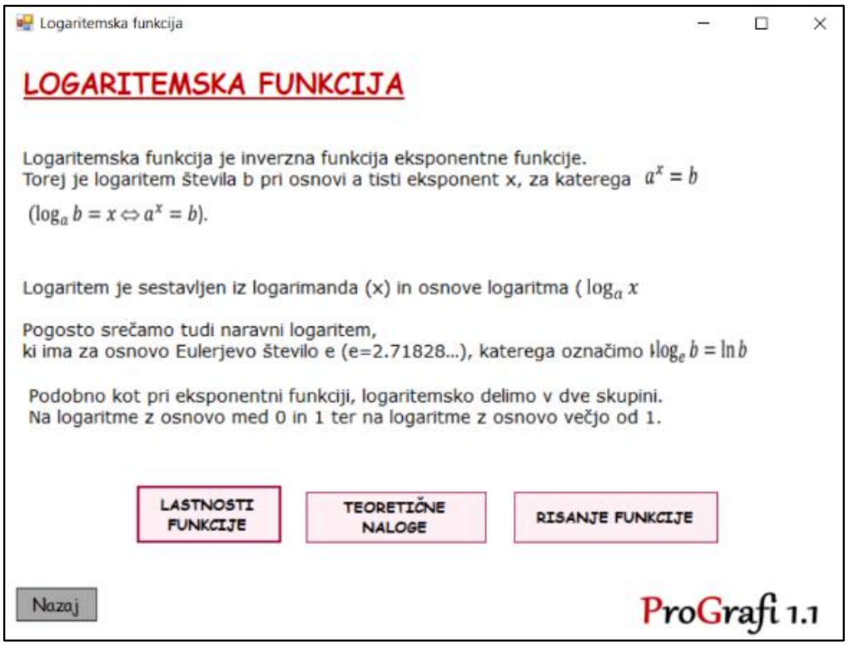
Ker predvidevam, da bi uporabnik rad preveril, če je eksponentno funkcijo narisal pravilno, vsebuje začetna stran tudi gumb »Risanje funkcije«, kamor lahko uporabnik vpiše dano enačbo in jo aplikacija tudi nariše (Slika 40).



Slika 40: Risanje eksponentne funkcije

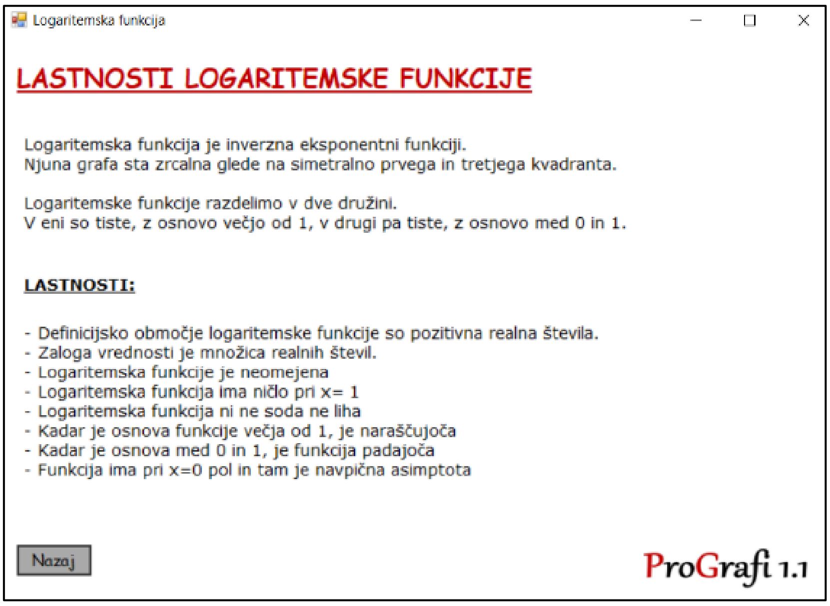
4.2.6 Logaritemska funkcija

Tudi pri logaritemski funkciji uporabnik najprej spozna definicijo le-te, nato pa se lahko odloči za reševanje različnih tipov nalog (Slika 41). Pod razlago se nahajajo trije gumbi, ki prikazujejo različne tipe nalog.



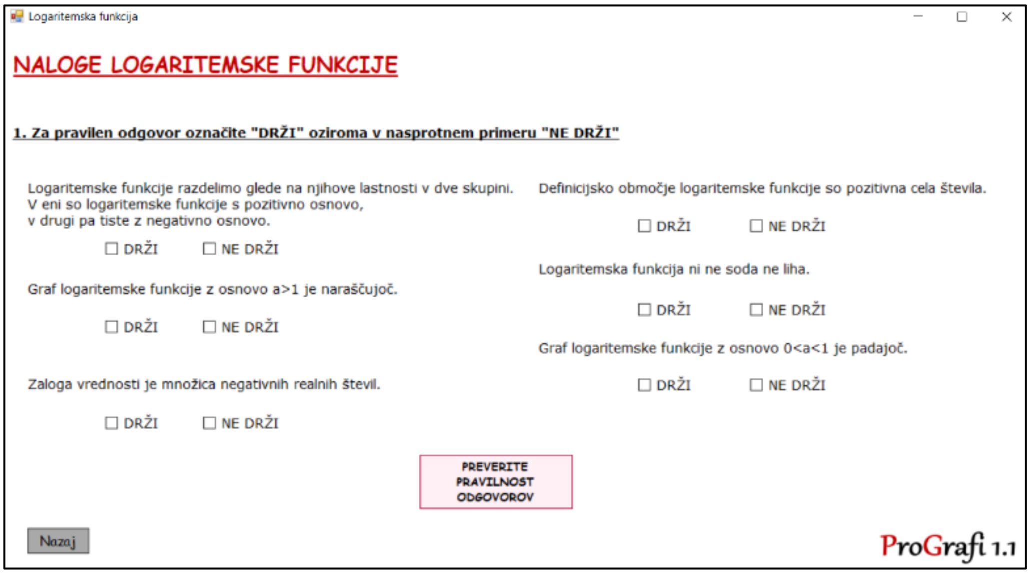
Slika 41: Začetna stran logaritemske funkcije

Ker napisana teorija na začetni strani ni dovolj, vsebuje logaritemska funkcija gumb »Lastnosti funkcije«, kjer je podrobneje opisano vse, kar mora uporabnik vedeti (Slika 42). Na začetku je obrazloženo, v katere skupine se logaritemska funkcija deli ter vse skupne lastnosti, ki so značilne za dano funkcijo.



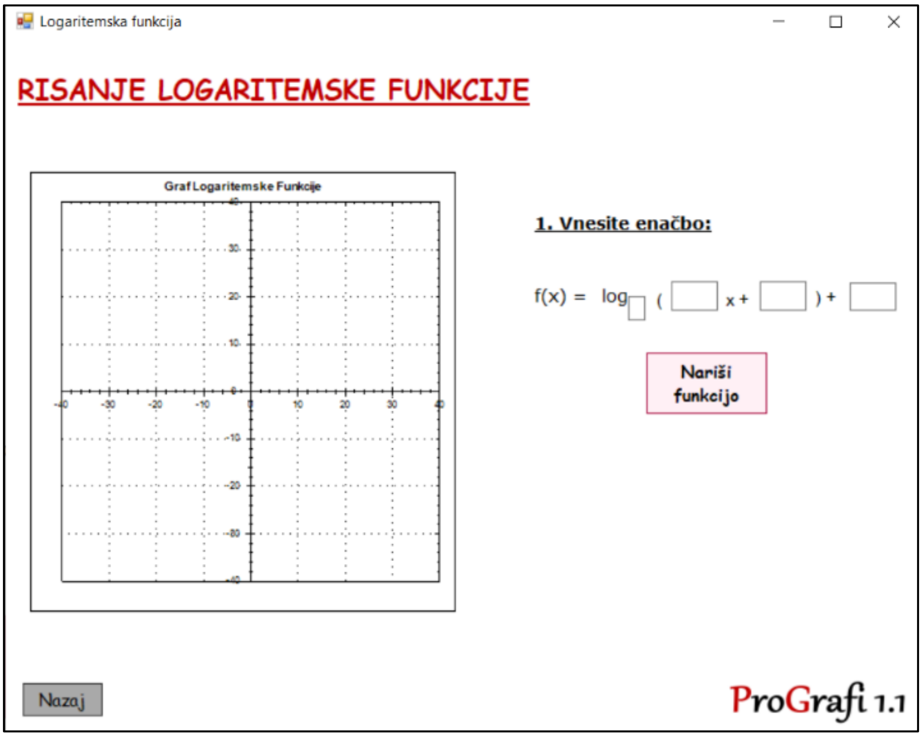
Slika 42: Lastnosti logaritemske funkcije

Po vsej teoriji lahko uporabnik tudi ponovi oz. obnovi teoretično znanje s pritiskom na gumb »Teoretične naloge«, kjer se odpre novo okence (Slika 43). V novem okencu se nahaja tip naloge, kjer mora uporabnik označiti, ali dana trditev drži ali ne drži. Prav tako lahko z gumbom »Preverite pravilnost odgovorov« uporabnik preveri, če je odgovore izbral pravilno.



Slika 43: Naloge logaritemske funkcije

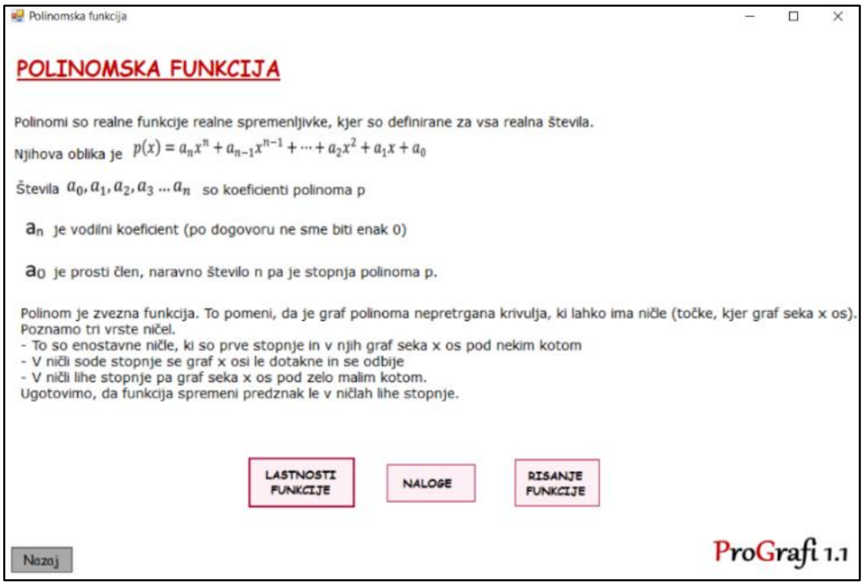
Zadnji gumb je »Risanje funkcije«, kjer uporabnik vnese dano enačbo in mu aplikacija glede na vnesene podatke nariše logaritemsko funkcijo (Slika 44). Tako lahko uporabnik preveri, če jo je narisal pravilno ali ne.



Slika 44: Risanje logaritemske funkcije

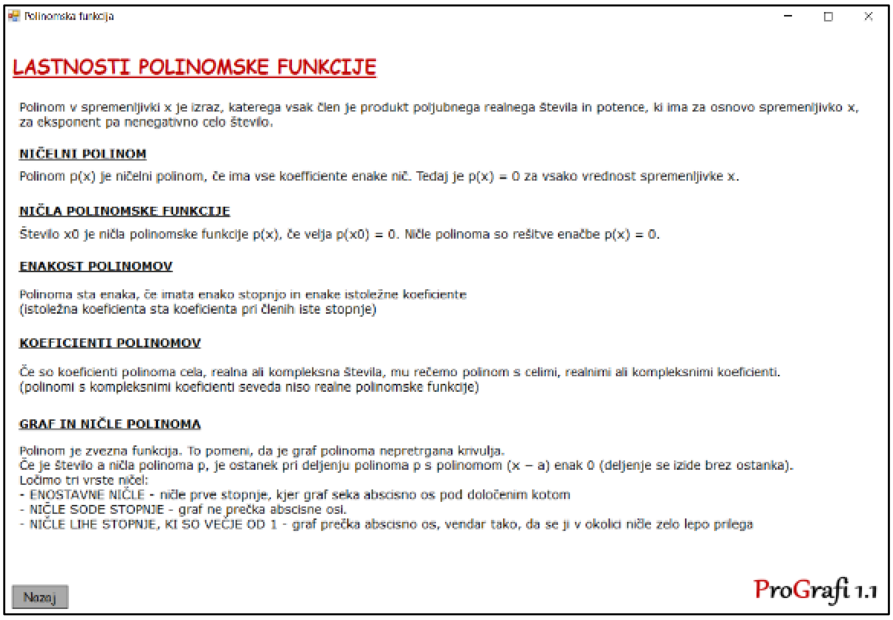
4.2.7 Polinomska funkcija

Polinomska funkcija je bila izbrana za eno težjih funkcij, zato tudi vsebuje razširjeno teorijo tako na začetni strani kot tudi v nadaljevanju. Po opisani teoriji sledijo trije gumbi z različnimi nalogami (Slika 45).



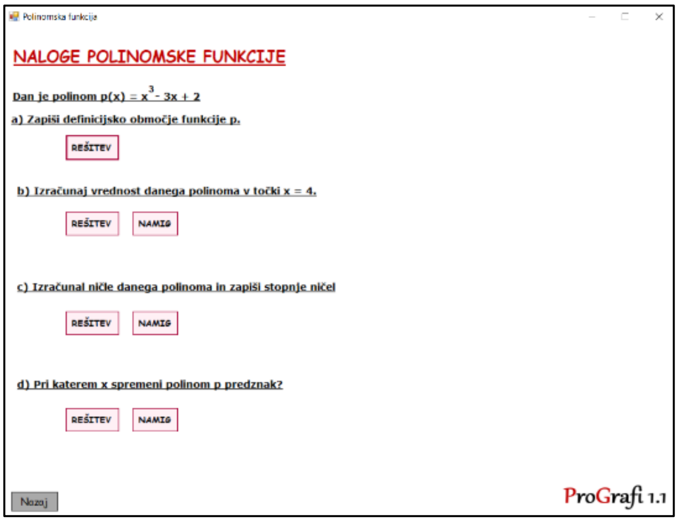
Slika 45: Začetna stran polinomske funkcije

Prvi gumb »Lastnosti funkcije«, kot sem že omenila, vsebuje lastnosti, ki naj bi jih uporabnik vedel, saj mu bodo pomagale pri reševanju različnih nalog. Opisani so ničelni polinomi, ničla polinomske funkcije, enakost in koeficienti polinomov ter graf in ničle polinoma (Slika 46).

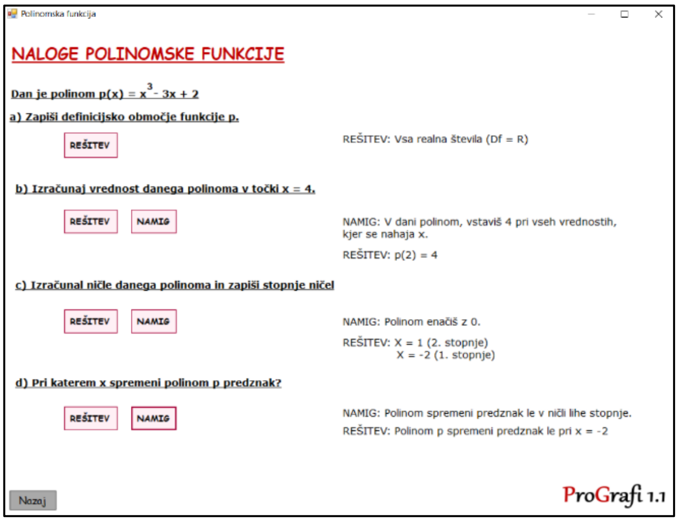


Slika 46: Lastnosti polinomske funkcije

Drugi gumb se imenuje »Naloge« in vsebuje razne tipske naloge, ki jih mora uporabnik sprva rešiti na list papirja (lahko tudi zvezek) (Slika 47). Če ima težave pri reševanju, ima vsak primer gumb »Namig«, na katerega uporabnik klikne in se prikaže pomoč, da lahko lažje reši nalogo. Ko konča z reševanjem primerov, lahko s klikom na gumb »Rešitev« preveri, ali je naloge rešil pravilno ali ne (Slika 48).

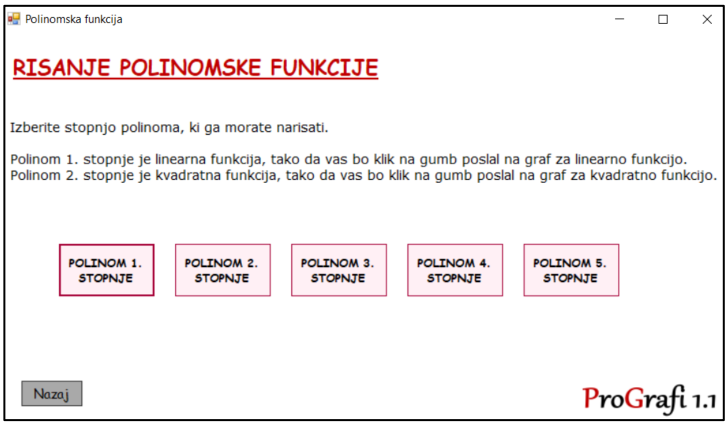


Slika 47: Naloge polinomske funkcije

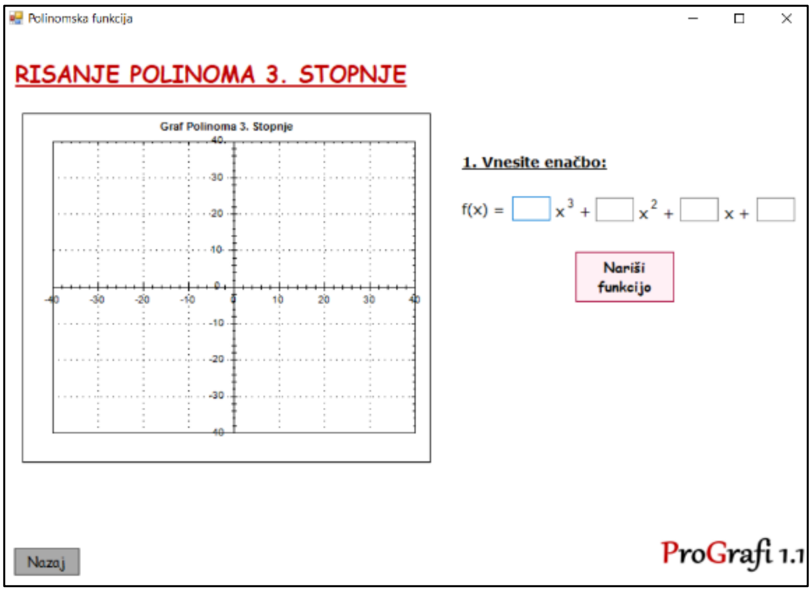


Slika 48: Naloge polinomske funkcije s vidnimi rešitvami in namigi

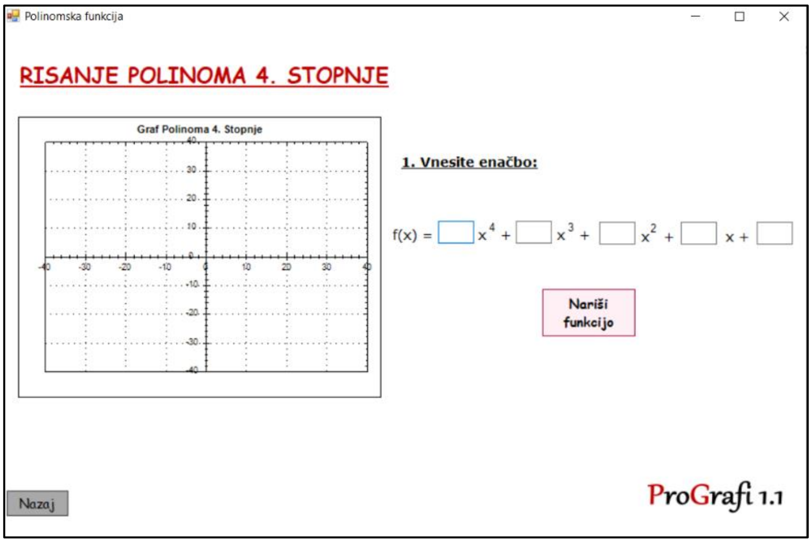
Polinomska funkcija vsebuje tudi gumb »Risanje funkcije«, kjer se prikaže okno s polinomi do 5. stopnje (Slika 49). Če uporabnik pritisne gumb za polinom 1. stopnje, se odpre okno za risanje linearne funkcije, saj je polinom 1. stopnje pravzaprav kar linearna funkcija. Podobno se zgodi, če uporabnik pritisne gumb za polinom 2. stopnje, saj se mu odpre okno za risanje kvadratne funkcije, ker je polinom 2. stopnje kvadratna funkcija. Če izbere gumb tretje (Slika 50), četrte (Slika 51) ali pete stopnje (Slika 52), se uporabniku odpre novo okno, kjer lahko vpiše enačbo polinoma izbrane stopnje in aplikacija glede na vneseno enačbo nariše graf danega polinoma.



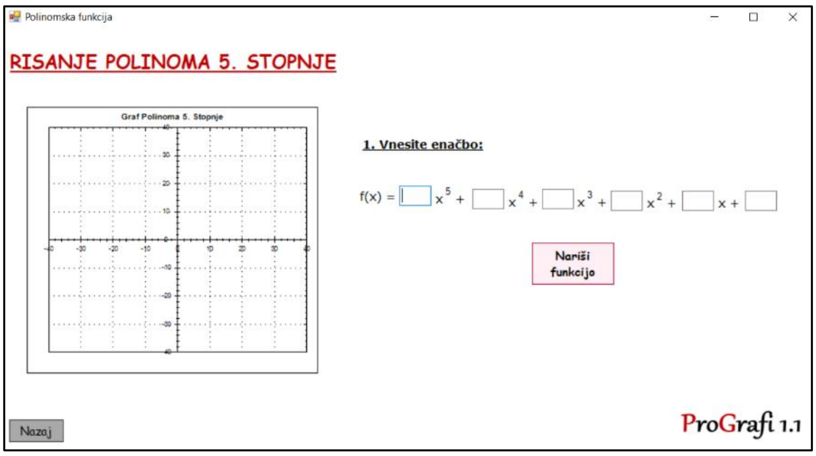
Slika 49: Risanje polinomske funkcije



Slika 50: Risanje polinoma 3. stopnje



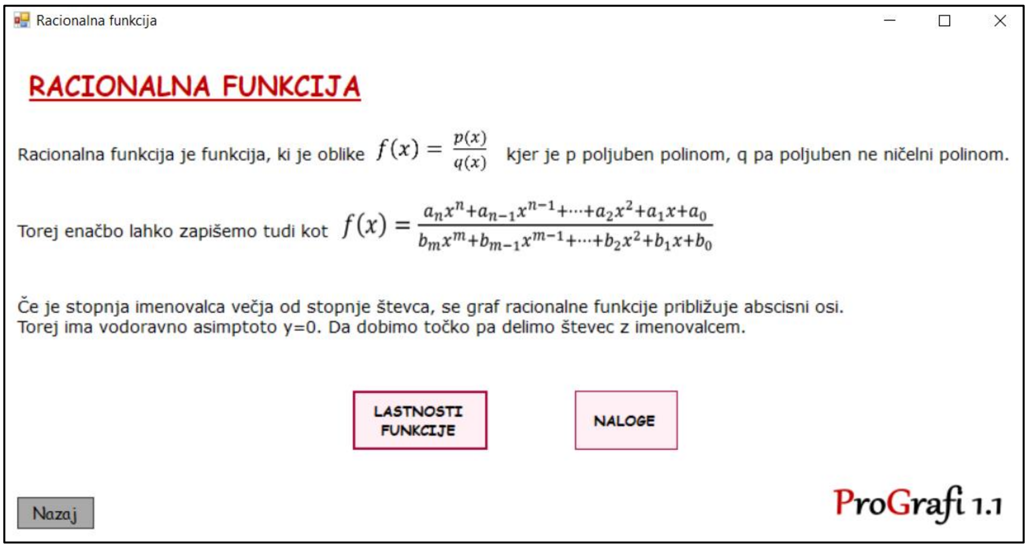
Slika 51: Risanje polinoma 4. stopnje



Slika 52: Risanje polinoma 5. stopnje

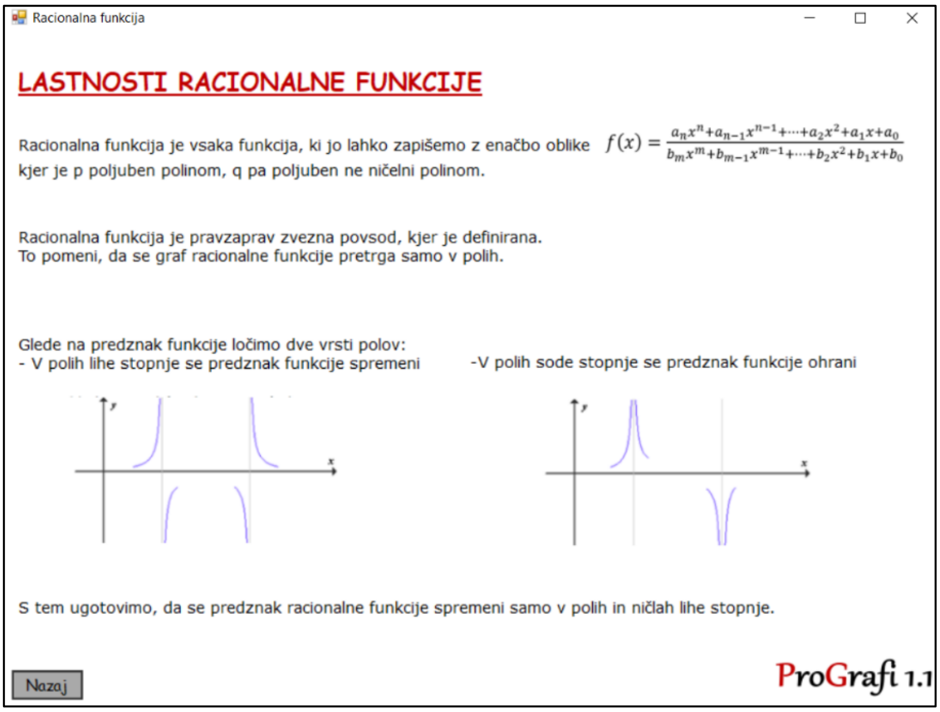
4.2.8 Racionalna funkcija

Ostane še racionalna funkcija. Začetna stran dane funkcije vsebuje kratko teorijo, da uporabnika seznani, kaj racionalna funkcija je (Slika 53). Pod napisano teorijo se nahajata dva gumba.



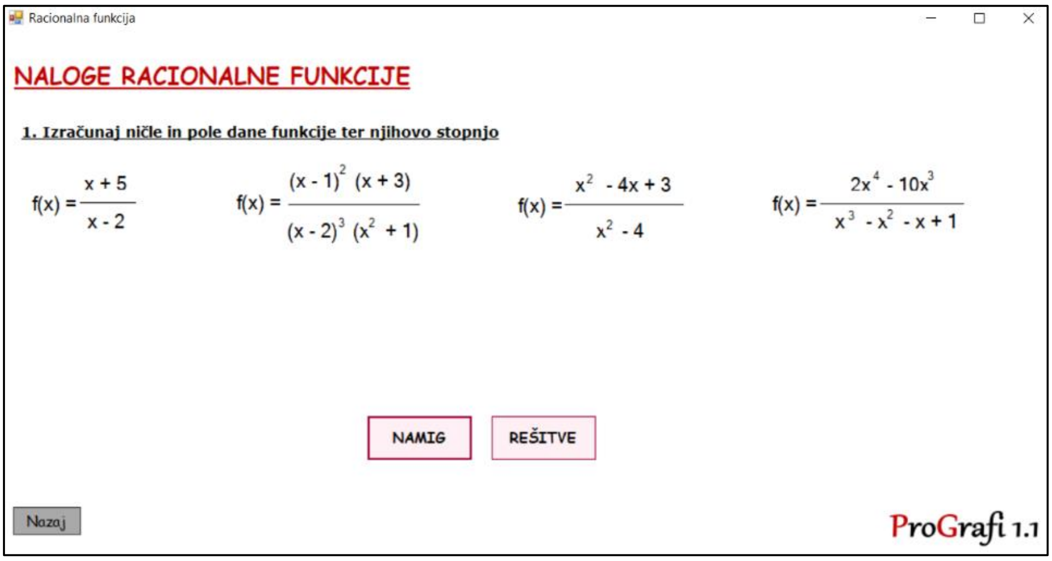
Slika 53: Začetna stran racionalne funkcije

Prvi gumb »Lastnosti funkcije« vsebuje podrobnejše lastnosti – priporočljivo je, da jih uporabnik pozna (Slika 54). Opisano je predvsem katere pole ločimo, kakšno obliko ima enačba ter kakšen je graf racionalne funkcije.

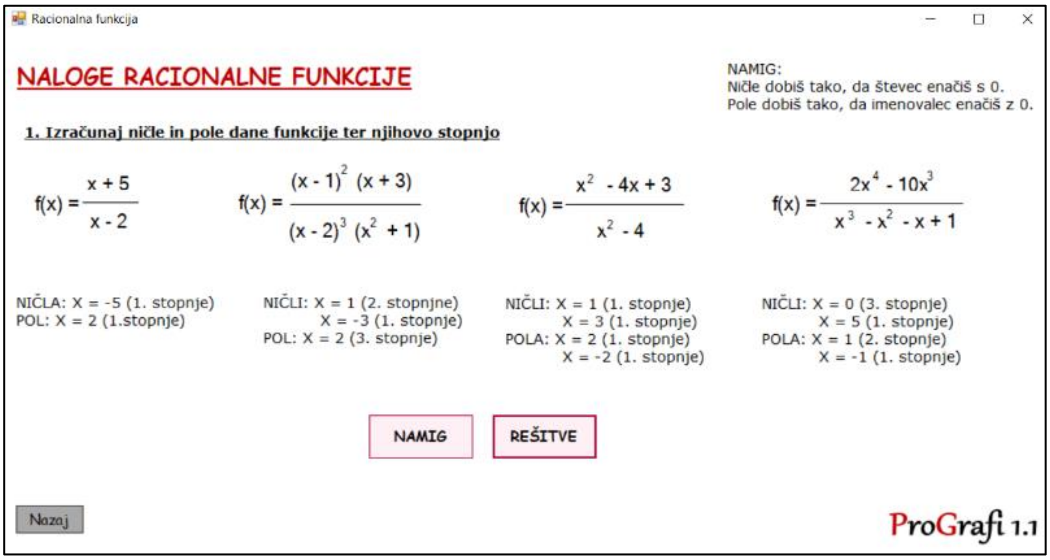


Slika 54: Lastnosti racionalne funkcije

Drugi gumb »Naloge« vsebuje tip naloge, kjer so uporabniku podane štiri racionalne funkcije, za katere mora sam izračunati ničle, pole ter določiti njihove stopnje (Slika 55). Če uporabnik ne ve, kako bi to izračunal, ima na dnu strani gumb »Namig«, na katerega klikne in se mu prikaže pomoč. Da pa lahko uporabnik preveri ali so njegovi odgovori pravilni, ima zraven še gumb »Rešitve« (Slika 56).



Slika 55: Naloge racionalne funkcije



Slika 56: Naloge racionalne funkcije s vidnimi rešitvami in namigom

4.3 Koda tipskih nalog

Kot je razvidno iz prejšnjega poglavja, je pri vsaki matematični funkciji, ki jo aplikacija ProGrafi 1.1 vsebuje, več različnih tipskih nalog. Ker ima veliko funkcij v aplikaciji enake vrste tipskih nalog, bom v nadaljevanju opisala kodo tipske naloge pri samo eni funkciji.

1. TIP – UPORABNIK VPIŠE ENAČBO PREMICE

Ta tip naloge se pojavi tako v kvadratni funkciji pri računanju pomembnih točk, kot v linearni funkciji pri računanju presečišča premice s koordinatnima osema.

Pri kvadratni funkciji program glede na vpisano enačbo izračuna presečišče z ordinatno osjo $Py(0, y)$, kjer je neznanka y sicer število c, ki je podano v enačbi. Nato izračuna točko T(p, q) oziroma teme s formulama $p = \frac{-b}{2a}$ in $q = \frac{-D}{4a}$. Za izračun q je neznanka diskriminanta (D), zato jo posebej izračuna po formuli $D = b^2 - 4ac$. Zadnjo pomembno presečišče, to sta ničli kvadratne funkcije, izračuna po formuli $x_{1,2} = \frac{-b \pm \sqrt{D}}{2a}$. Ker pa ničli ne obstajata vedno, vsebuje koda še "if stavke", s katerimi preverja, če je D večji, manjši ali enak 0. Če je $D > 0$, sta obe ničli kvadratne funkcije realni, če je $D = 0$, sta števili x_1 in x_1 enaki, torej ima kvadratna funkcija samo eno, dvojno realno ničlo, če je $D < 0, pa$ sta obe ničli kvadratne funkcije kompleksni, kar pomeni, da graf funkcije ne seka abscisne osi v realnem koordinatnem sistemu. Postopek računanja pomembnih presečišč parabole lahko vidite na Sliki 57.

```
1 reference
private void Button1_Click(object sender, EventArgs e) // gumb Izračunaj
{
    double p, q, px, py, d, x1, x2;
    double a, b, c;
    double.TryParse(textBox_A.Text, out a);
    double.TryParse(textBox_B.Text, out b);
    double.TryParse(textBox_C.Text, out c);
    //Izračuna D
    d = (Math.Pow(b, 2)) - (4 * a * c);
    if (d >0)
    {
        x1 =( ((-b) + Math.Sqrt(d))) /( 2 * a);
        x1 = Math.Round(x1,2);
        x2 = (((-b) - Math.Sqrt(d))) / (2 * a);
        x2 = Math.Round(x2, 2);
        textBox_Nicli.Text = "x1 = " + x1.ToString() + ", x2 = " + x2.ToString();

    }
    if (d==0)
    {
        x1 = ((-b) - Math.Sqrt(d)) / 2 * a;
        x1 = Math.Round(x1,2);
        textBox_Nicli.Text = "ničla je samo ena - " + x1.ToString();
    }
    if (d<0)
    {
        textBox_Nicli.Text = "ničli ne obstajata";
    }
    p = -(b) / (2 * a);
    q = (-d) / (4 * a);
    py = c;
    px = 0;
    //Izpiše vrednosti
    textBox_P.Text = "T("+ p.ToString() + ", " + q.ToString()+ ")";
    textBox_Px.Text ="Py(" + px.ToString() + ", " + py.ToString() + ")";
}
```

Slika 57: Koda računanja pomembnih presečišč parabole

Pri linearni funkciji program glede na dane podatke izračuna vsako presečišče posebej. Za točko $N(0, y)$ potrebuje samo še y, ki ga dobi pri ničli funkciji, saj graf tam seka ordinatno os. Za točko $M(X, O)$ potrebuje program samo še x, ki ga izračuna po formuli $x = -\frac{n}{k}$. Graf nariše tako, da program za vsak x izračuna vrednost y po formuli $y = kx + n$ s tako imenovanim tabeliranjem. Postopek računanja presečišč lahko vidite na Sliki 58.

```
1 reference
private void Button1_Click(object sender, EventArgs e) // gumb izračunaj presečišče točk
{
    double k, n, x;
    double.TryParse(TextBox_K.Text, out k);
    double.TryParse(TextBox_n.Text, out n);
    double y = 0;
    x = -(n / k);
    TextBox_M.Text = "(" + x.ToString() + ", " + y.ToString() + ")";
    TextBox_N2.Text = "(" + y.ToString() + ", " + n.ToString() + ")";
}
```

Slika 58: Koda računanja presečišč

2. TIP – UPORABNIK VPIŠE KOORDINATE DVEH TOČK

Ta tip naloge se pojavi samo pri linearni funkciji. Pomembno je, da uporabnik vpiše obe koordinati pri obeh točkah, saj drugače program ne bo deloval, ker ne bo prejel nobenih podatkov. Program glede na koordinate dveh točk, ki jih vpiše uporabnik, izračuna enačbo premice, ki poteka skozi dve podani točki, ki ju je vnesel uporabnik. Ker sta smerni koeficient (k) in začetna vrednost (n) neznana, ju program izračuna s formulami $k = \frac{x_2 - x_1}{y^2 - y_1}$. in $n = y_1 - kx_1$. Aplikacija graf nariše tako, da program za vsak x izračuna vrednost y po formuli $y = kx + n$ s tako imenovanim tabeliranjem. Postopek računanja enačbe lahko vidite na Sliki 59.

```
1 reference
private void Button2_Click(object sender, EventArgs e) // gumb Napiši enačbo
{
    // izračuna vrednosti k in n
    double x1, x2, y1, y2, k, n;
    double.TryParse(TextBox_X1.Text, out x1);
    double.TryParse(TextBox_X2.Text, out x2);
    double.TryParse(TextBox_Y1.Text, out y1);
    double.TryParse(TextBox_Y2.Text, out y2);
    k = (y2 - y1) / (x2 - x1);
    n = y1 - k * x1;

    //izpiše premico
    TextBox_EnacbaPremice.Text = "f(x) = " + Math.Round(k, 3, MidpointRounding.ToEven) + " x + " + Math.Round(n, 3, MidpointRounding.ToEven);
}
```

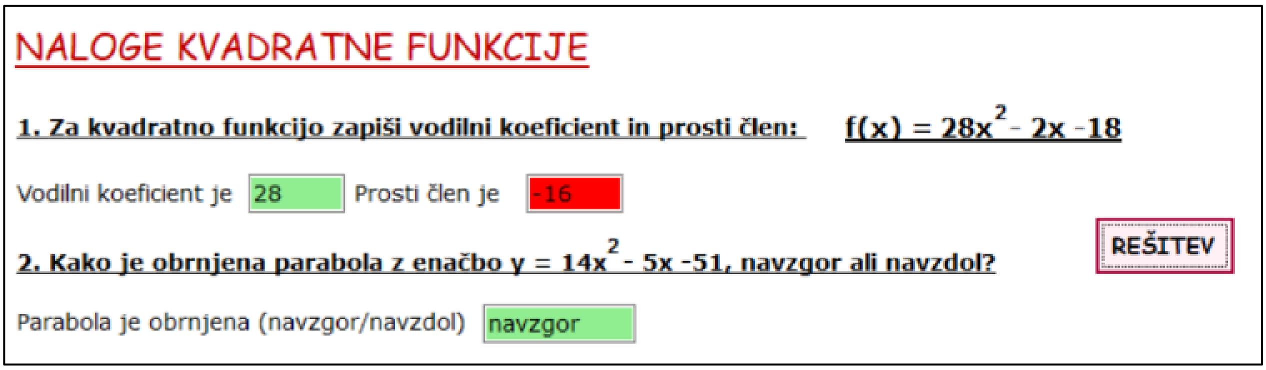
Slika 59: Koda za računanje linearne enačbe

3. TIP – UPORABNIK VPIŠE BESEDILO V TEXTBOX

Ta tip naloge se pojavi pri linearni, kvadratni in potenčni funkciji. Pomembno je, da uporabnik vpiše besedo oziroma številko in ne pusti prazno, saj drugače program ne bo deloval. Vpisuje v gradnik, ki se imenuje TextBox. Koda deluje tako, da vpisani rezultat preverja s pravilnim If-pogojem. Če je odgovor pravilno napisan, se kvadratek obarva zeleno. Če pa uporabnik ne vpiše pravilnega rezultata, se kvadratek obarva rdečo. To sem dosegla z Else-pogojem. Ker je pri vseh zgoraj omenjenih funkcijah isto, sem za primer izbrala nalogo pri kvadratni funkciji. Na Sliki 60 vidite kodo postopka, na Sliki 61 pa lahko vidite obarvane kvadratke, kadar so pravilni oz. nepravilni.

```
1 reference
private void button2_Click(object sender, EventArgs e)
{
    if (textBox1.Text == "28") { textBox1.BackColor = Color.LightGreen; } //odgovor pravilen
    else { textBox1.BackColor = Color.Red; } //odgovor ne pravilen

    if (textBox2.Text == "-18") { textBox2.BackColor = Color.LightGreen; }//odgovor pravilen
    else { textBox2.BackColor = Color.Red; }//odgovor ne pravilen

    if (textBox3.Text == "navzgor") { textBox3.BackColor = Color.LightGreen; }//odgovor pravilen
    else { textBox3.BackColor = Color.Red; }//odgovor ne pravilen
}
```

Slika 60: Koda za preverjanje pravilnega odgovora



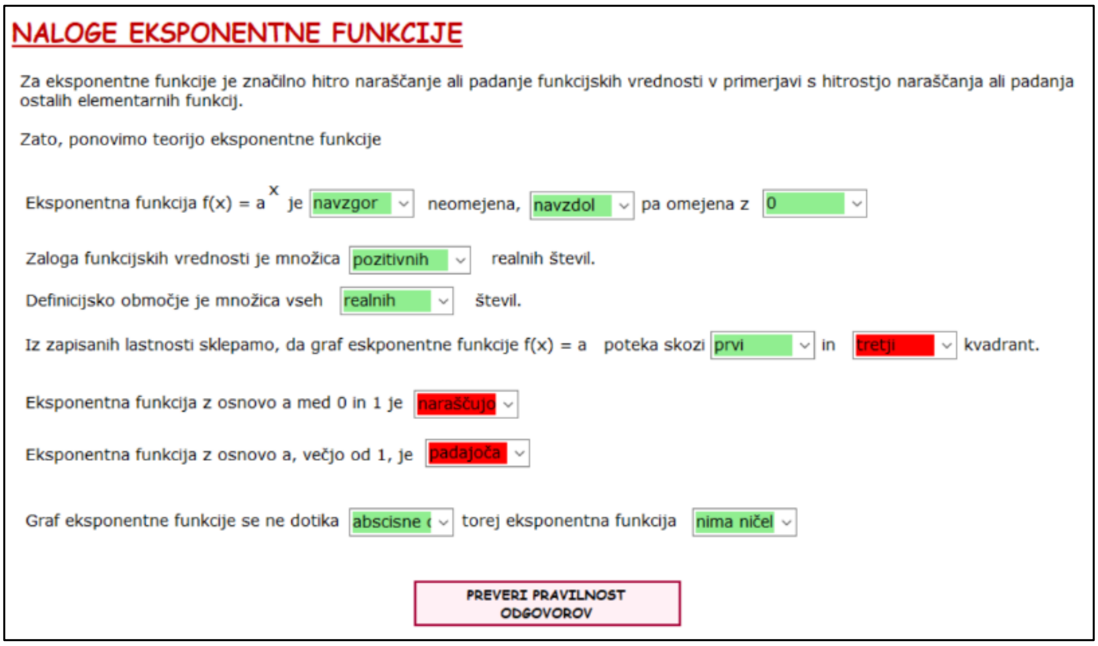
Slika 61: Rešene naloge kvadratne funkcije

4. TIP – UPORABNIK IZBERE ENO IZMED PODANIH MOŽNOSTI

Ta tip naloge se pojavi v aplikaciji največkrat, in sicer v kvadratni, linearni, potenčni, korenski ter eksponentni funkciji. Je najbolj priročna oblika, saj ima uporabnik različne odgovore že podane in samo izbere pravilnega. Ni težav, da bi uporabnik kakšno črko napačno zapisal ali katero izmed črk zapisal z veliko oz. malo začetnico. Za primer kode bom uporabila naloge eksponentne funkcije, saj je celotna naloga sestavljena iz tega tipa nalog. Vse dane izbire odgovorov so zapisane v gradniku, ki se imenuje ComboBox, izbire se pa imenujejo elementi. Element, ki ga izbere uporabnik, se preveri z If-stavkom, če je enak pravilnemu odgovoru. Če je odgovor pravilen, se izbrani rezultat obarva z zeleno barvo. Z Else-stavkom pa aplikacija ugotovi, da je odgovor nepravilen. Nepravilen odgovor oz. izbrani odgovor se tako obarva rdeče barve. Na Sliki 62 lahko vidite kodo postopka preverjanja, na Sliki 63 pa lahko vidite obarvane izbire, kadar so pravilne oz. nepravilne.

```
1 reference
private void button1_Click(object sender, EventArgs e)
{
    if (comboBox1.SelectedItem.ToString() == "navzgor"){comboBox1.BackColor = Color.LightGreen;} //izbira je pravilna
    else{comboBox1.BackColor = Color.Red;} //izbira je nepravilna

    if (comboBox2.SelectedItem.ToString() == "navzdol") { comboBox2.BackColor = Color.LightGreen; }//izbira je pravilna
    else { comboBox2.BackColor = Color.Red; }//izbira je nepravilna

    if (comboBox3.SelectedItem.ToString() == "0") { comboBox3.BackColor = Color.LightGreen; }//izbira je pravilna
    else { comboBox3.BackColor = Color.Red; }//izbira je nepravilna

    if (comboBox4.SelectedItem.ToString() == "pozitivnih") { comboBox4.BackColor = Color.LightGreen; }//izbira je pravilna
    else { comboBox4.BackColor = Color.Red; }//izbira je nepravilna

    if (comboBox5.SelectedItem.ToString() == "realnih") { comboBox5.BackColor = Color.LightGreen; }//izbira je pravilna
    else { comboBox5.BackColor = Color.Red; }//izbira je nepravilna

    if (comboBox6.SelectedItem.ToString() == "prvi") { comboBox6.BackColor = Color.LightGreen; }//izbira je pravilna
    else { comboBox6.BackColor = Color.Red; }//izbira je nepravilna

    if (comboBox7.SelectedItem.ToString() == "drugi") { comboBox7.BackColor = Color.LightGreen; }//izbira je pravilna
    else { comboBox7.BackColor = Color.Red; }//izbira je nepravilna

    if (comboBox8.SelectedItem.ToString() == "padajoča") { comboBox8.BackColor = Color.LightGreen; }//izbira je pravilna
    else { comboBox8.BackColor = Color.Red; }//izbira je nepravilna

    if (comboBox9.SelectedItem.ToString() == "naraščujoča") { comboBox9.BackColor = Color.LightGreen; }//izbira je pravilna
    else { comboBox9.BackColor = Color.Red; }//izbira je nepravilna

    if (comboBox10.SelectedItem.ToString() == "abscisne osi") { comboBox10.BackColor = Color.LightGreen; }//izbira je pravilna
    else { comboBox10.BackColor = Color.Red; }//izbira je nepravilna

    if (comboBox11.SelectedItem.ToString() == "nima ničel") { comboBox11.BackColor = Color.LightGreen; }//izbira je pravilna
    else { comboBox11.BackColor = Color.Red; }//izbira je nepravilna

}
```

Slika 62: Koda za preverjanje pravilnosti nalog



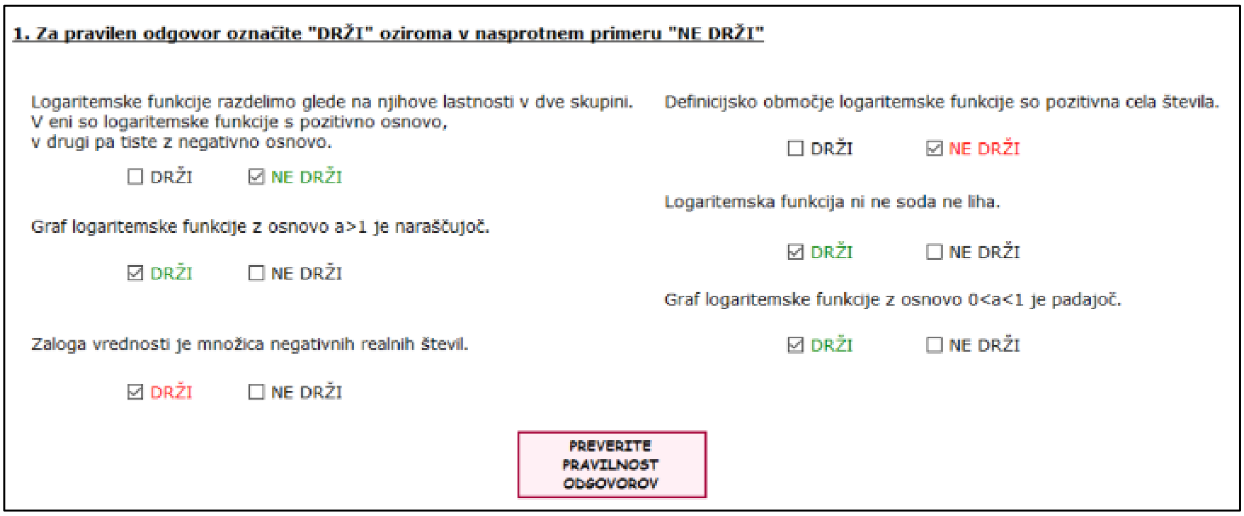
Slika 63: Rešene naloge eksponentne funkcije

5. TIP – UPORABNIK OBKLJUKA PRAVO REŠITEV (DRŽI/NE DRŽI, VELJA/NE VELJA)

Ta tip naloge se pojavi pri korenski in logaritemski funkciji. Uporabnik mora obkljukati tisto izbiro, za katero misli, da je pravilna (drži ali velja). Gradnik, katerega uporabnik označuje, se imenuje CheckBox. Za primer bom izbrala nalogo iz logaritemske funkcije, saj je celotna naloga sestavljena iz tega tipa. Za vsak izbrani CheckBox se izvede If-stavek, ki preveri, kateri CheckBox se je izbral. Če se je izbral prvi (drži), se besedilo obarva zeleno. Če pa se je izbral drugi (ne drži), pa se besedilo obarva rdeče. Na Sliki 64 lahko vidite kodo, na Sliki 65 pa besedila obarvana zelene oz. rdeče barve.

```
1 reference
private void button1_Click(object sender, EventArgs e)
{
    if (checkBox1.Checked) { checkBox1.ForeColor = Color.Red; } //DRŽI --> NEPRAVILEN ODGOVOR
    if (checkBox2.Checked) { checkBox2.ForeColor = Color.Green; } // NE DRŽI --> PRAVILEN ODGOVOR

    if (checkBox3.Checked) { checkBox3.ForeColor = Color.Green; } //DRŽI --> PRAVILEN ODGOVOR
    if (checkBox4.Checked) { checkBox4.ForeColor = Color.Red; } // NE DRŽI --> NEPRAVILEN ODGOVOR

    if (checkBox5.Checked) { checkBox5.ForeColor = Color.Red; } //DRŽI --> NEPRAVILEN ODGOVOR
    if (checkBox6.Checked) { checkBox6.ForeColor = Color.Green; } // NE DRŽI --> PRAVILEN ODGOVOR

    if (checkBox7.Checked) { checkBox7.ForeColor = Color.Green; } //DRŽI --> PRAVILEN ODGOVOR
    if (checkBox8.Checked) { checkBox8.ForeColor = Color.Red; } // NE DRŽI --> NEPRAVILEN ODGOVOR

    if (checkBox9.Checked) { checkBox9.ForeColor = Color.Green; } //DRŽI --> PRAVILEN ODGOVOR
    if (checkBox10.Checked) { checkBox10.ForeColor = Color.Red; } // NE DRŽI --> NEPRAVILEN ODGOVOR

    if (checkBox11.Checked) { checkBox11.ForeColor = Color.Green; } //DRŽI --> PRAVILEN ODGOVOR
    if (checkBox12.Checked) { checkBox12.ForeColor = Color.Red; } // NE DRŽI --> NEPRAVILEN ODGOVOR
}
```

Slika 64: Koda za preverjanje pravilnost odgovorov



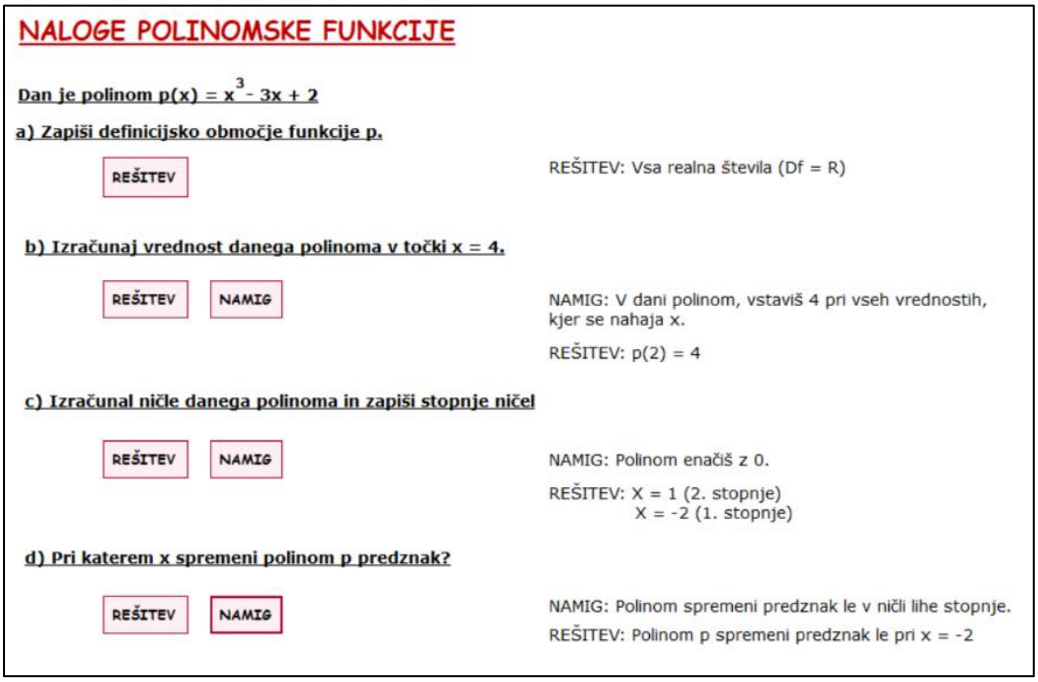
Slika 65: Rešene naloge logaritemske funkcije

6. TIP – UPORABNIK PRITISNE GUMB ZA NAMIG IN REŠITEV

Ta tip naloge se pojavi pri polinomski in racionalni funkciji. Pri tej nalogi je uporabniku podana naloga, ki jo mora rešiti sam (na pamet, na list, v zvezek) ter s klikom na gumb »Rešitev« preveri pravilnost. Kadar pa uporabnik ne zna rešiti naloge, ima možnost namiga, ki mu pomaga pri reševanju. Pritisniti mora samo na gumb »Namig«. Kot primer bom izbrala nalogo iz polinomske funkcije. Vsa besedila so napisana v gradniku z imenom Label. Besedilo, kjer sta napisana namig in rešitev, nista vidna uporabniku. Ko pa uporabnik pritisne določen gumb, ki ga potrebuje, se mu besedilo prikaže. To sem dosegla z ukazom *label.Visible=true*. Kodo lahko vidite na Sliki 66, prikaz rešitev in namiga pa lahko vidite na Sliki 67.

```
1 reference
private void button1_Click(object sender, EventArgs e) //GUMB REŠITEV
{
    label7.Visible = true; //rešitev a) primera
}
1 reference
private void button2_Click(object sender, EventArgs e) //GUMB REŠITEV
{
    label8.Visible = true; //rešitev b) primera
}
1 reference
private void button6_Click(object sender, EventArgs e) //GUMB NAMIG
{
    label9.Visible = true; //namig b) primera
}
1 reference
private void button3_Click(object sender, EventArgs e) //GUMB REŠITEV
{
    label10.Visible = true; //rešitev c) primera
}
1 reference
private void button7_Click(object sender, EventArgs e) //GUMB NAMIG
{
    label11.Visible = true; //namig b) primera
}
1 reference
private void button5_Click(object sender, EventArgs e) //GUMB REŠITEV
{
    label12.Visible = true; //rešitev d) primera
}
1 reference
private void button8_Click(object sender, EventArgs e) //GUMB NAMIG
{
    label13.Visible = true; //namig b) primera
}
```

Slika 66: Koda za vidnost rešitev in namigov



Slika 67: Naloge polinomske funkcije s rešitvami in namigi

4.4 Koda risanja funkcij

Večina matematičnih funkcij, ki jih vsebuje aplikacija ProGrafi 1.1, ima tip naloge, kjer uporabnik vpiše enačbo premice in mu aplikacija nato na podlagi vpisane enačbe nariše potrebni graf. Ker pa je koda povsod narejena na enak način, bom za primer izbrala samo eno funkcijo, in sicer polinomsko funkcijo 4. stopnje. Za risanje funkcij sem izbrala knjižnico ZedGraph, ki je podrobneje opisana v poglavju 2.3.

Za pričetek je potrebno nastaviti koordinatni sistem in imenovanje koordinatnih osi ter grafa. Pri nastavljanju grafa je bilo potrebno nastaviti, da se osi sekata v središču S(0,0), to sem dosegla z metodo *myPane.YAxis.Cross oz. myPane.XAxis.Cross*. Prav tako sem hotela, da se prikaže mreža koordinatnega sistema – pri tem mi je pomagala metoda *myPane.YAxis.MajorGrid.IsVisiable* oz. *myPane.XAxis.MajorGrid.IsVisiable*. Potrebno je bilo še določiti velikost koordinatnega sistema. Pri vseh funkcijah se je koordinatni sistem prikazal pri točkah 40 oz. – 40. Kodo opisanega lahko vidite na Sliki 68.

```
1 reference
private void plotGraph()
{
    GraphPane myPane = zedGraphControl1.GraphPane;
    myPane.Title.Text = "Graf Polinoma 4. Stopnje"; //ime grafa
    myPane.XAxis.Title.Text = " ";
    myPane.YAxis.Title.Text = " ";

    PointPairList krnekiPairList = new PointPairList();

    //središče obeh osi pri 0
    myPane.YAxis.Cross = 0;
    myPane.XAxis.Cross = 0;

    // mreža vidna pri x in y osi
    myPane.YAxis.MajorGrid.IsVisible = true;
    myPane.XAxis.MajorGrid.IsVisible = true;

    //dolžina koordinatnega sistema na začetku
    myPane.YAxis.Scale.Min = -40;
    myPane.YAxis.Scale.Max = 40;
    myPane.XAxis.Scale.Min = -40;
    myPane.XAxis.Scale.Max = 40;

    zedGraphControl1.AxisChange();

}
```

Slika 68: Nastavitev koordinatnega sistema

Sedaj ko sem določila vse potrebno pri koordinatnem sistemu, je bilo potrebno sam graf še narisati. Tega sem se lotila s tako imenovanim tabeliranjem, kjer sem y vrednosti že imela podane z vpisano uporabnikovo enačbo, potrebno je bilo samo še izračunati vrednosti x. Program je v enačbo, kjer se nahaja vrednost x, vpisoval vrednosti od – 1000 do 1000, s korakom po 0,1.

Torej če vzamem za primer polinomsko funkcijo 4. stopnje. Uporabnik je v enačbo $ax^4 + bx^3 + cx^2 + dx + e$, vpisal vrednosti a, b, c, d in e. Program je nato glede na dano vpisano enačbo izračunal vsak x vrednosti y po zgoraj dani formuli s tako imenovanim tabeliranjem. Kodo za naslavljanje grafa ter tabeliranje lahko vidite na Sliki 69.

```
1 reference
private double[] YPODATKI() //NASLAVLJANJE GRAFA
{
    // izračuna vrednosti Y in jih shranjuje v polje
    double[] ypodatki = new double[20000];
    double x, y;
    double a, b, c, d, e;
    double.TryParse(textBox_A.Text, out a);
    double.TryParse(textBox_B.Text, out b);
    double.TryParse(textBox_C.Text, out c);
    double.TryParse(textBox_D.Text, out d);
    double.TryParse(textBox_E.Text, out e);
    int i = 0;

    for (x = -1000; x < 1000; x = x + 0.1)
    {
        y = a * Math.Pow(x, 4) + b * Math.Pow(x, 3) + c * Math.Pow(x, 2) + d * x + e;

        ypodatki[i] = y;
        i++;
    }
    return ypodatki;
}

1 reference
private void button1_Click_1(object sender, EventArgs e)
{
    // nariše premico
    PointPairList krnekiPairList = new PointPairList();
    GraphPane myPane = zedGraphControl1.GraphPane;

    double[] data1 = YPODATKI();
    int j = 0;
    for (double i = -1000; i < 1000; i = i + 0.1) //TABELIRANJE
    {
        krnekiPairList.Add(i, data1[j]);
        j++;
    }
    LineItem krnekiACurve = myPane.AddCurve("graf1", krnekiPairList, Color.Red, SymbolType.None); // nariše graf
}
```

Slika 69: Primer naslavljanja in tabeliranja grafa polinoma 4. stopnje

5 TEŽAVE PRI USTVARJANJU APLIKACIJE

Glavni cilj je bila nadgradnja lanske aplikacije, ki sem jo hkrati naredila še priročnejšo ter lepšo za ogled. Določila sem, da bo aplikacija risala vse matematične elementarne funkcije, vendar sem naletela na težavo pri korenski in racionalni funkciji.

Težava pri racionalni funkciji se pojavi pri tako imenovanih polih oz. kjer funkcija ni definirana, tam mi skrajni dve točki poveže, čemur seveda ni tako. Problema sem se lotila tako iz matematičnega kot iz programerskega vidika, vendar mi težave ni uspelo rešiti.

Pri korenski funkciji sem imela težavo s programskim delom. Risanja sem se lotila s pretvorbo na eksponent, saj v Windows Forms Application ni bilo pravega modula. Zaradi tega aplikacija ni risala lihih korenov. Problem so predstavljali tudi kubični koreni, čeprav sem koren spremenila na eksponent, aplikacija še vedno ni risala pravilno. Zaradi tako ozke izbire, ki bi jo program lahko ponudil, sem se tudi pri tej funkciji odločila, da je aplikacija ne bo risala.

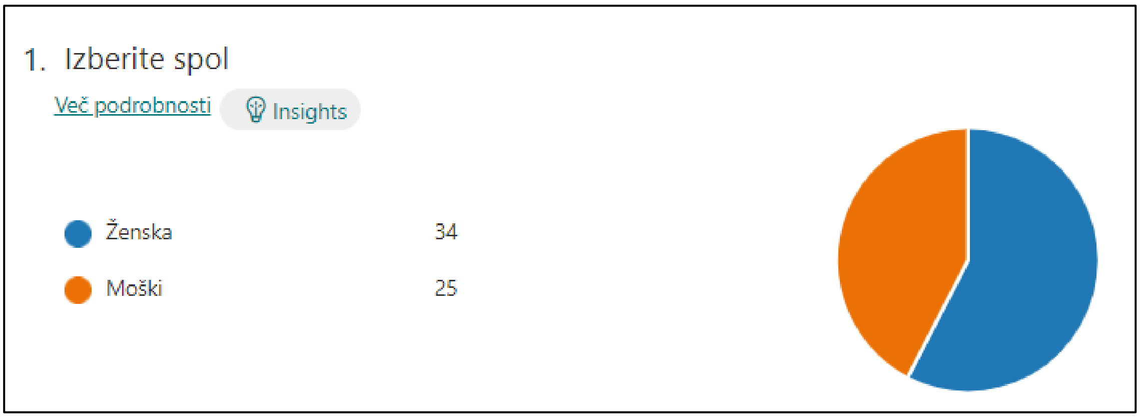
Pri zasnovi tipskih nalog sem na začetku naletela na problem, saj nisem bila prepričana, katere naloge bi uporabila, da bi bila aplikacija čim uporabnejša. Problem sem rešila z anketo, saj sem s tem dobila vpogled v realno stanje pri dijakih.

Ko je aplikacija začela delovati pravilno, sem se lotila oblikovnega dela. Videz lanske verzije ProGrafi mi ni bil všeč, zato sem ga hotela nadgraditi. Naletela pa sem na težavo, in sicer Windows Forms Application nima najbolj prijaznega uporabniškega vmesnika in je precej omejen. Ker pa nisem imela dovolj časa, da bi spremenila platformo in pričela od začetka, sem se morala spopasti s to težavo. Uporabljala sem razne slike enačb, besedila drugačne pisave, dodala logotip, itd.

6 ANALIZA REZULTATOV ANKETE

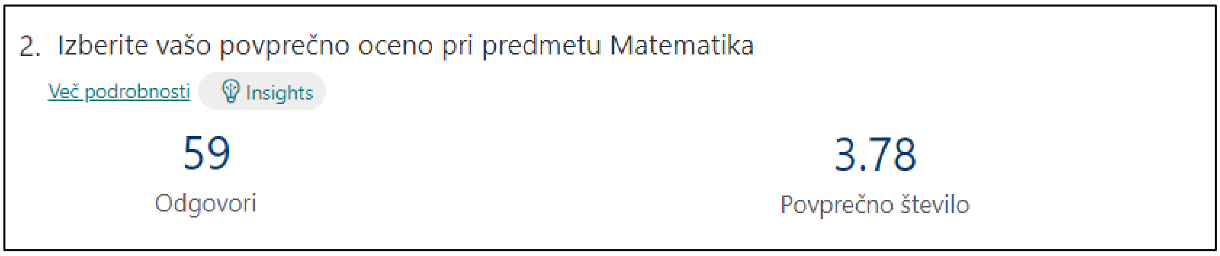
Ker sem imela manjše probleme pri določanju tipskih nalog pri določenih funkcijah, sem se odločila, da bom raziskala, katere matematične funkcije povzročajo dijakom največ težav in katere tipske naloge bi najraje reševali (risanje grafov ali računanje nalog). Glede na rezultate sem se lahko lotila sestavljanju nalog pri funkcijah, ki jih aplikacija ProGrafi 1.1 nudi.

Anketo je rešilo 59 dijakov. Glede na spol je anketo rešilo več žensk, in sicer 34 (57,63 %), medtem ko je bilo anketirancev moškega spola 25 (42,37 %).



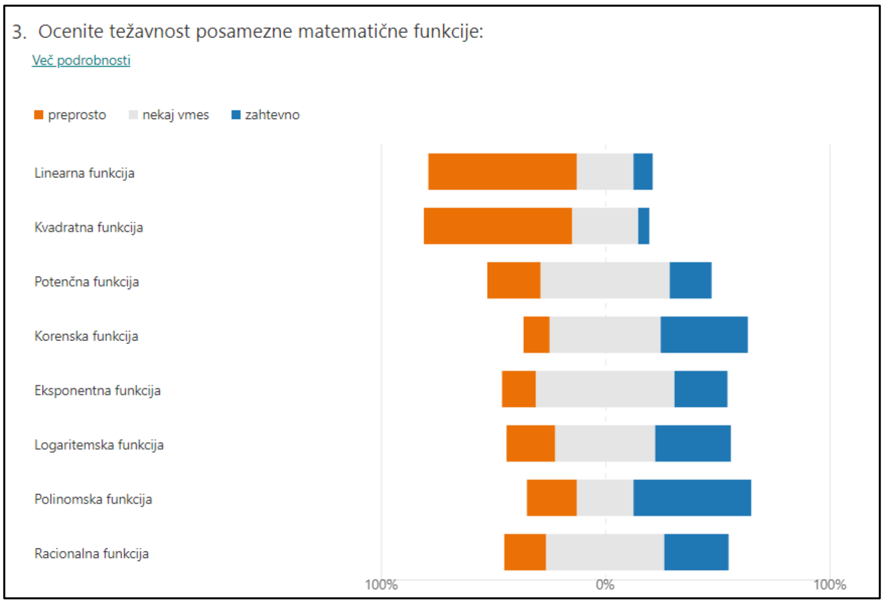
Graf 1: Izberite spol.

Pri prvem vprašanju sem anketirance vprašala, kakšno povprečno oceno imajo pri predmetu matematika. Imeli so možnosti od ena do pet. Povprečno število vseh odgovor je bilo 3.78.



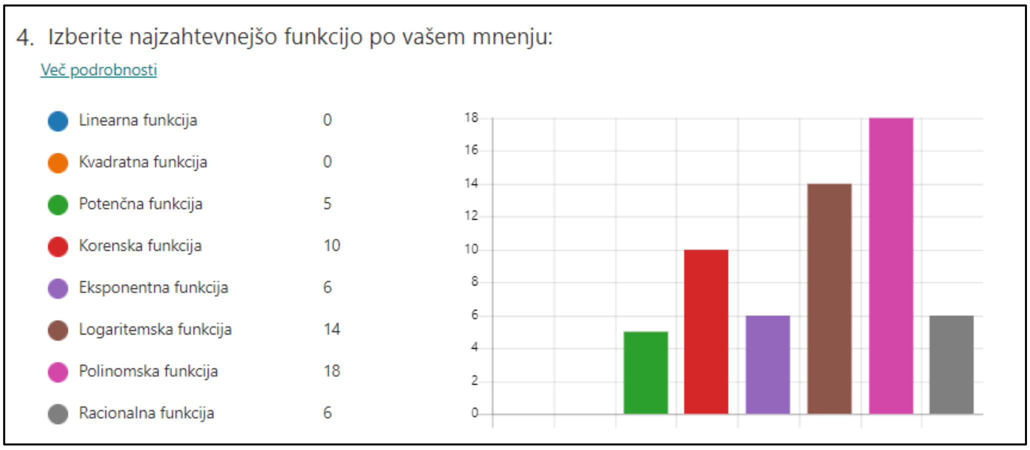
Graf 2: Izberite vašo povprečno oceno pri predmetu matematika

Pri drugem vprašanju me je zanimalo, kako bodo dijaki ocenili težavnost posameznih matematičnih funkcij, ki jih aplikacija ProGrafi 1.1 vsebuje. Rezultati so bili precej raznoliki. Anketirancem je najtežja funkcija polinomska, saj jo je kot zahtevno označilo 52,5 % oseb. Po zahtevnosti ji nato sledi korenska funkcija, kjer jo za zahtevno označilo 39 %. Med preprostejše funkcije sodita linearna in kvadratna funkcija, saj sta bili s kar 66.1 % označeni kot preprosti. Vmes pa sodijo eksponentna, logaritemska ter racionalna funkcija.



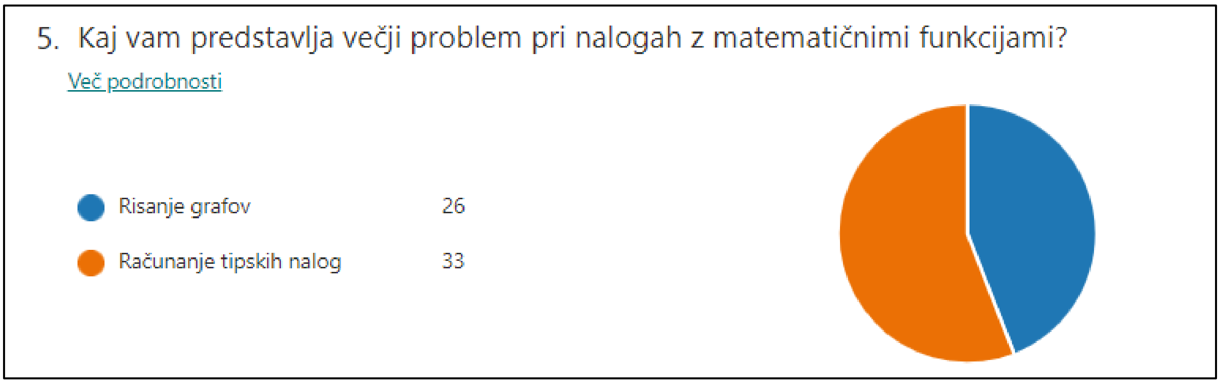
Graf 3: Ocenite težavnost posamezne matematične funkcije.

Pri četrtem vprašanju sem anketirance spraševala, katera funkcija je po njihovem mnenju najzahtevnejša. Za najzahtevnejšo funkcijo je 18 oseb izbralo polinomsko funkcijo, nato ji sledi logaritemska funkcija s 14 označbami. Kot najpreprostejši funkciji sta linearna in kvadratna, saj med 59 anketiranci nista bili izbrani.



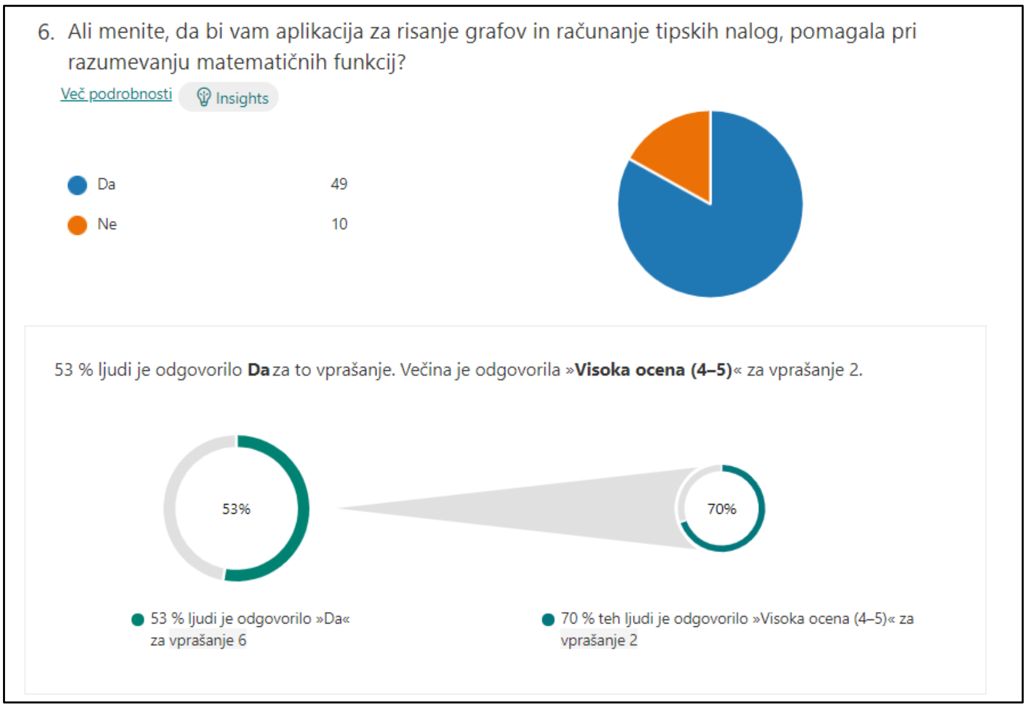
Graf 4: Izberite najzahtevnejšo funkcijo po vašem mnenju.

Pri petem vprašanju me je zanimalo, kateri tip naloge bi bil dijakom uporabnejši oziroma jim predstavlja večji problem. Kot možnost so imeli na izbiro risanje grafov in računanje tipskih nalog. 33 osebam večji problem predstavlja računanje tipskih nalog, na drugi strani pa kar 26 osebam predstavlja problem risanje grafov. Zaradi tega sem v aplikaciji ProGrafi poskusila vključiti oba tipa v večino funkcij.



Graf 5: Kaj vam predstavlja večji problem pri nalogah z matematičnimi funkcijami?

Pri zadnjem vprašanju sem anketirance vprašala, če bi jim aplikacija, ki riše grafe ter računa tipske naloge, pomagala pri razumevanju matematičnih funkcij. Kar 83 % oseb je odgovorilo z »Da«. Visok procent mi je veliko pomenil, saj to pomeni, da ima aplikacija dejanski smisel.



Graf 6: Ali menite, da bi vam aplikacija za risanje grafov in računanje tipskih nalog pomagala pri razumevanju matematičnih funkcij?

7 UGOTOVITVE

Na začetku sem postavila 5 hipotez, v nadaljevanju so opisane ugotovitve.

H1: Matematični del bo zame predstavljal večjo težavo kot programski. Hipoteze ne morem potrditi, saj je programski del predstavljal več problemov kot matematični.

H2: Anketa bo pripomogla pri nadgradnji programa. Hipotezo sem potrdila, saj je anketa zelo pripomogla pri sestavljanju in določanju tipskih nalog.

H3: Srednješolsko znanje ne zadostuje za izdelavo aplikacije. Hipotezo sem potrdila, saj aplikacija ProGrafi 1.1 ne riše določenih funkcij zaradi pomanjkanja znanja.

H4: Anketiranci bodo za najtežjo funkcijo izbrali Racionalno funkcijo. Hipoteze ne morem potrditi, saj so anketiranci za najtežjo funkcijo izbrali polinomsko funkcijo.

H5: Aplikacija bo risala vse elementarne funkcije, ki se obravnavajo v srednji šoli. Hipoteze ne morem potrditi, saj aplikacija ProGrafi 1.1 ne riše korenske in racionalne funkcije.

8 ZAKLJUČEK

Za raziskovalno nalogo sem morala povezati znanje z veliko različnih področij, predvsem pa iz matematike in računalništva. Zadovoljna sem, da mi je uspelo narediti medpredmetno povezavo in nadgraditi svoje znanje tako matematike kot programiranja, saj sem morala do veliko spoznanj priti sama s pomočjo različnih gradiv. Nadgradila sem razumevanje matematike v računalništvu, razumevanje posameznih funkcij in implementacijo le-teh v program. Spoznala sem se s kreiranjem nalog in nenazadnje tudi s samim pisanjem programa. Prišla sem do spoznanja, da je razvijanje aplikacije, ki je dejansko namenjena uporabniku, precej dolgotrajen proces in da ko rešiš eno težavo, že takoj naletiš na drugo. Verjamem, da mi bo takšno razmišljanje pri nadaljevanju šolanja zelo pomagalo.

Naletela sem na kar nekaj težav. Večino sem jih rešila, nekatere pa so ostale za prihodnje verzije programa. V aplikacijo je bilo vloženega veliko prostega časa in sem vesela, da mi je uspelo.

9 VIRI IN LITERATURA

SPATIUM Novum: Matematika za gimnazije. 1. izdaja, 3. natis. Ljubljana: Modrijan založba. 2015.

PLANUM Novum: Matematika za gimnazije. 1. izdaja. Ljubljana: Modrijan založba. 2012.

BRILEJ R., SELJAK R. in ŠPEGEL RAZBORNIK A. ALFA 4: Zbirka nalog za matematiko v 4. letniku srednjega tehniškega oz. strokovnega izobraževanja. ATAJA. Ljubljana: ATAJA. 2004.

KRANER, M. Program za delo s funkcijami: raziskovalna naloga. (2020).

C-sharpcorner.com. Pridobljeno 21. februar 2021 iz Using CheckBox in Windows Forms: https://www.c-sharpcorner.com/article/working-with-checkbox-control-in-winforms-application2/.

Docs.microsoft.com. Pridobljeno 21. februar 2021 iz ComboBox Class: https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.combobox?view=net-5.0.

Osvoji znanje. Pridobljeno 21. februar 2021 iz https://osvojiznanje.weebly.com/matematika1/lastnosti-matematicne-funkcije.

Scott Lilly. Pridobljeno 12. januar 2020 iz https://scottlilly.com/learn-c-by-building-a-simple-rpg-index/lesson-00-3-the-parts-of-visual-studio/.

ZedGraph. Pridobljeno 12. januar 2020 iz http://zedgraph.sourceforge.net/samples.html.

Večina slik je iz lastnega arhiva. Slike pridobljene s spleta, imajo poleg imena še povezavo, na kateri so dostopne.

10 PRILOGE

ANKETA

Program za delo s funkcijami

Pozdravljeni,
sem Maša Kraner, dijakinja 4. letnika srednje strokovne šole za računalništvo.
Pri raziskovalni nalogi bom nadgradila aplikacijo ProGrafi, ki računa posamezne tipe nalog in riše različne matematične elementarne funkcije, ki se obravnavajo v srednji šoli. Prosila bi vas, da si vzamete nekaj minut časa in rešite anketo, saj bi mi s tem zelo pomagali pri nadgradnji aplikacije.
Hvala že vnaprej za vašo sodelovanje in odgovore!

1. Izberite spol *

- Ženska
- Moški

2. Izberite vašo povprečno oceno pri predmetu Matematika *

1	2	3	4	5
○	○	○	○	○

3. Ocenite težavnost posamezne matematične funkcije: *

	preprosto	nekaj vmes	zahtevno
Linearna funkcija	○	○	○
Kvadratna funkcija	○	○	○
Potenčna funkcija	○	○	○
Korenska funkcija	○	○	○
Eksponentna funkcija	○	○	○
Logaritemska funkcija	○	○	○
Polinomska funkcija	○	○	○
Racionalna funkcija	○	○	○

4. Izberite najzahtevnejšo funkcijo po vašem mnenju: *

- Linearna funkcija
- Kvadratna funkcija
- Potenčna funkcija
- Korenska funkcija
- Eksponentna funkcija
- Logaritemska funkcija
- Polinomska funkcija
- Racionalna funkcija

5. Kaj vam predstavlja večji problem pri nalogah z matematičnimi funkcijami? *

- Risanje grafov
- Računanje tipskih nalog

6. Ali menite, da bi vam aplikacija za risanje grafov in računanje tipskih nalog, pomagala pri razumevanju matematičnih funkcij? *

- Da
- Ne

11 IZJAVA

Mentor Boštjan Resinovič, v skladu z 2. in 17. členom Pravilnika raziskovalne dejavnosti »Mladi za Celje« Mestne občine Celje, zagotavljam, da je v raziskovalni nalogi z naslovom Program za delo s funkcijami, katere avtorica je Maša Kraner:

- besedilo v tiskani in elektronski obliki istovetno,
- pri raziskovanju uporabljeno gradivo navedeno v seznamu uporabljene literature,
- da je za objavo fotografij v nalogi pridobljeno avtorjevo dovoljenje in je hranjeno v šolskem arhivu,
- da sme Osrednja knjižnica Celje objaviti raziskovalno nalogo v polnem besedilu na knjižničnih portalih z navedbo, da je raziskovalna naloga nastala v okviru projekta Mladi za Celje,
- da je raziskovalno nalogo dovoljeno uporabiti za izobraževalne in raziskovalne namene s povzemanjem misli, idej, konceptov oziroma besedil iz naloge ob upoštevanju avtorstva in korektnem citiranju,
- da smo seznanjeni z razpisni pogoji projekta Mladi za Celje.

ŠOLSKI CENTER CELJE
SREDNJA ŠOLA ZA KEMIJO, ELEKTROTEHNIKO IN RAČUNALNIŠTVO
1
CELJE, Pot na Lavo 22

Celje, 9.5.2021

žig šole:

Podpis mentorja:

Podpis odgovorne osebe: