Izrada C# aplikacije za pretraživanje grafova

Doko, Petar

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Preddiplomski sveučilišni studij računarstva

Izrada C# aplikacije za pretraživanje grafova

Završni rad

Petar Doko

Osijek, 2016.

ETFOS
ELEKTROTEHNIČKI FAKULTET OSIJEK
Sveučilište Josipa Jurja Strossmayera u Osijeku

Obrazac Z1P - Obrazac za ocjenu završnog rada na preddiplomskom studiju

Osijek,

Odboru za završne i diplomske ispite

Prijedlog ocjene završnog rada

Ime i prezime studenta:	Petar Doko
Studij, smjer:	Računarstvo
Mat. br. studenta, godina upisa:	R-3590, 2013.
Mentor:	doc.dr.sc. Ivan Aleksi
Sumentor:	
Naslov završnog rada:	C# APLIKACIJA ZA PRETRAŽIVANJE GRAFOVA
Primarna znanstvena grana rada:	Računarstvo
Sekundarna znanstvena grana (ili polje) rada:	
Predložena ocjena završnog rada:	
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: Postignuti rezultati u odnosu na složenost zadatka: Jasnoća pismenog izražavanja: Razina samostalnosti:

Potpis sumentora:	Potpis mentora:

Dostaviti:

1. Studentska služba

Potpis predsjednika Odbora: _____

Dostaviti:

1. Studentska služba

ELEKTROTEHNIČKI FAKULTET OSIJEK
Sveučilište Josipa Jurja Strossmayera u Osijeku

IZJAVA O ORIGINALNOSTI RADA

Osijek,

Ime i prezime studenta:	Petar Doko
Studij :	Računarstvo
Mat. br. studenta, godina upisa:	R-3590, 2013.

Ovom izjavom izjavljujem da je rad pod nazivom**:**

C# APLIKACIJA ZA PRETRAŽIVANJE GRAFOVA

izrađen pod vodstvom mentora

doc.dr.sc. Ivan Aleksi

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

SADRŽAJ

1. UVOD

Cilj ovog završnog rada je izraditi aplikaciju za pretraživanje grafova. Za izradu aplikacije koristi se C# programski jezik u Visual Studiju 2015. Na samome početku trebalo je steći određena predznanja na ovome području rada, a ta saznanja su stećena na kolegijima „Objektno-orijentirano programiranje“ i „Programiranje 2“. Bitna stavka u ovoj aplikaciji je dodavanje čvorova na formi u C# jer se sa njima onda obavljaju sve ostale funkcije. Korisnik aplikacije vidi samo gotovu aplikaciju, odnosno njen grafički prikaz. Korisnik u ovoj aplikaciji može dodavati i brisati čvorove, mijenjati njihove koordinate te odabrati neku od metoda pretraživanja po želji. Programski kod koji to sve obavlja je nevidljiv krajnjem korisniku, on vidi samo grafički dio(sučelje) aplikacije.

1.1 Zadatak završnog rada

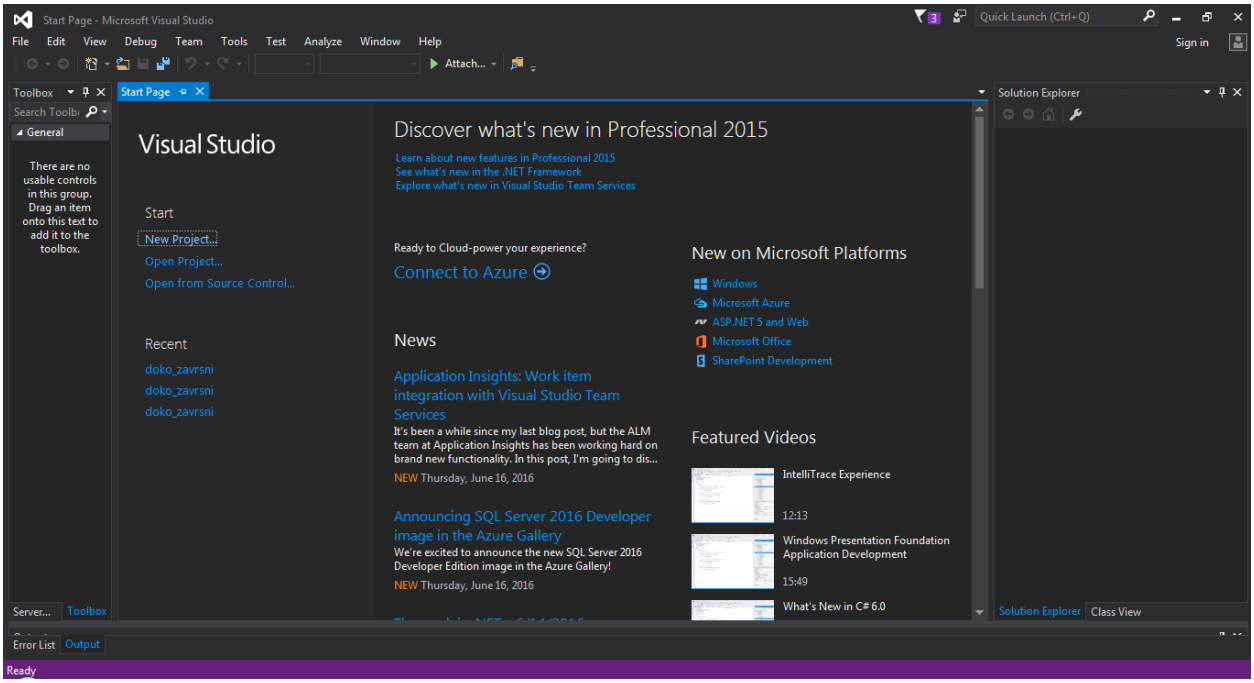
Zadatak ovog završnog rada je napraviti C# aplikaciju za pretraživanje grafova. Potrebno je omogućiti dodavanje/brisanje čvorova, postavljanje koordinata, metode pretraživanja i prikaz vremena pretraživanja, udaljenost start-cilj te broj pretraženih čvorova.

2. PRIMIJENJENI ALATI

2.1 Microsoft Visual Studio

Microsoft Visual Studio je integrirano razvojno okruženje(IDE) dizajniran od strane Microsoft-a. Koristi se za razvoj računalnih programa za Windows, web stranice, aplikacije i usluge. Visual Studio sadrži urednik izvornog koda koji podržava IntelliSense (dodatak koja predlaže nadopunu koda) kao i promjenu koda. Osim IntelliSense sadrži i debugger koji radi na razini izvornog i strojnog koda. Program također sadrži alate poput dizajnera oblika koji se koristi za pravljenje aplikacija sa grafičkom korisničkim sučeljem, web dizajnera, dizajnera klasa i dizajnera shema baza podataka. Prihvaća proširenja koja poboljšavaju funkcionalost na skoro svakom nivou dodavajući podršku sistema za upravljanje izvornim kodom (poput subversion softvera) i dodajući nove setove alata poput tekstualnih urednika i vizualnih dizajnera za jezik specifičnih domena ili za druge dijelove procesa razvoja softvera

Visual Studio podržava različite programske jezike i dozvoljava uredniku koda i debuggeru da podržava skoro bilo koji programski jezik, pod uvjetom da postoji servis za taj jezik. Neki od ugrađenih jezika su C, C++, C++/CLI(preko Visual C++), VB.NET, C#(preko Visual C#) i F#. Instalacijom različitih servisa mogu se podržati jezici poput M, Phyton i Ruby. Također podržava XML/XSLT, HTML/XHTML, JavaScript i CSS.

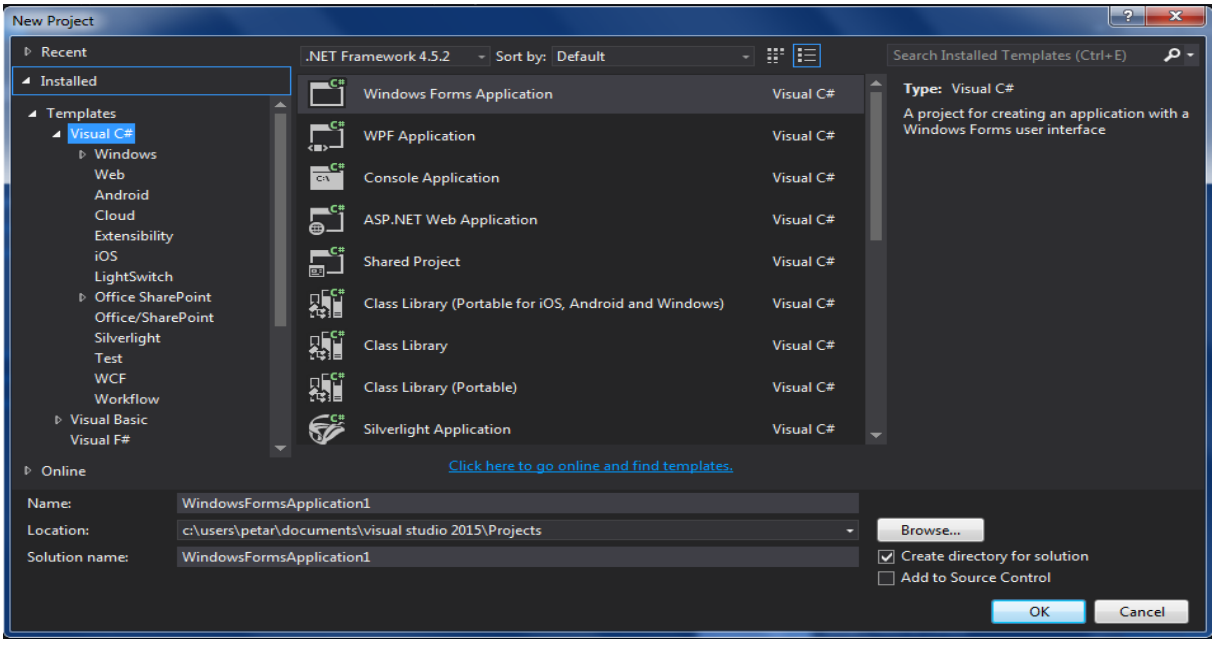


Slika 2.1 Prikaz početne stranice Microsoft Visual Studio 2015.

2.2 C#

U ovome radnom okruženju korišten je programski jezik C# za izradu ovoga rada. C# je objektno orijentirani programski jezik visokih performansi za .NET platformu, razvijen od tvrtke Microsoft s ciljem da bude jednostavan, siguran, moderan i jezik opće namjene. Na tržištu se pojavio 2000. godine, a nastao je na temelju objektno orijentiranih jezika C++ i Visual Basic. Sintaksa i semantika su preuzete iz Java programskog jezika, a grafički objekti se temelje na Visual Basic-u. C# ima brojne mogućnosti vezane za definiranje klasa i metoda, kao i korištenju enkapsulacije, nasljeđivanja i polimorfizma. Podržava XML stil unutar dokumenata, sučelja, svojstva, događaje te podržava rad s pokazivačima i prikupljanje smeća (eng. *garbage collection*) koji omogućuje oslobađanje memorije od objekata koji se više ne koriste, što znači da o tome ne mora brinuti programer.

Tipovi podataka koji se mogu koristiti u C# jeziku su također slični poput C i C++ jezika. Osnovni tipovi su char, int, short, long, float, double, bool, string, decimal. Osim tih tipova postoje i izvedeni tipovi, kao što su naprimjer uint, ushort i ulong. Svaki od tih tipova sprema određeni tip podatka i zauzima određenu količinu memorije u računalu. Za ispis na konzolu koristi se naredba Console.WriteLine(„ispis teksta“);. Ako želimo od korisnika preuzeti neki upis s tipkovnice i spremiti ga u varijablu tekst, to možemo napraviti idućom naredbom string tekst = Console.ReadLine();.

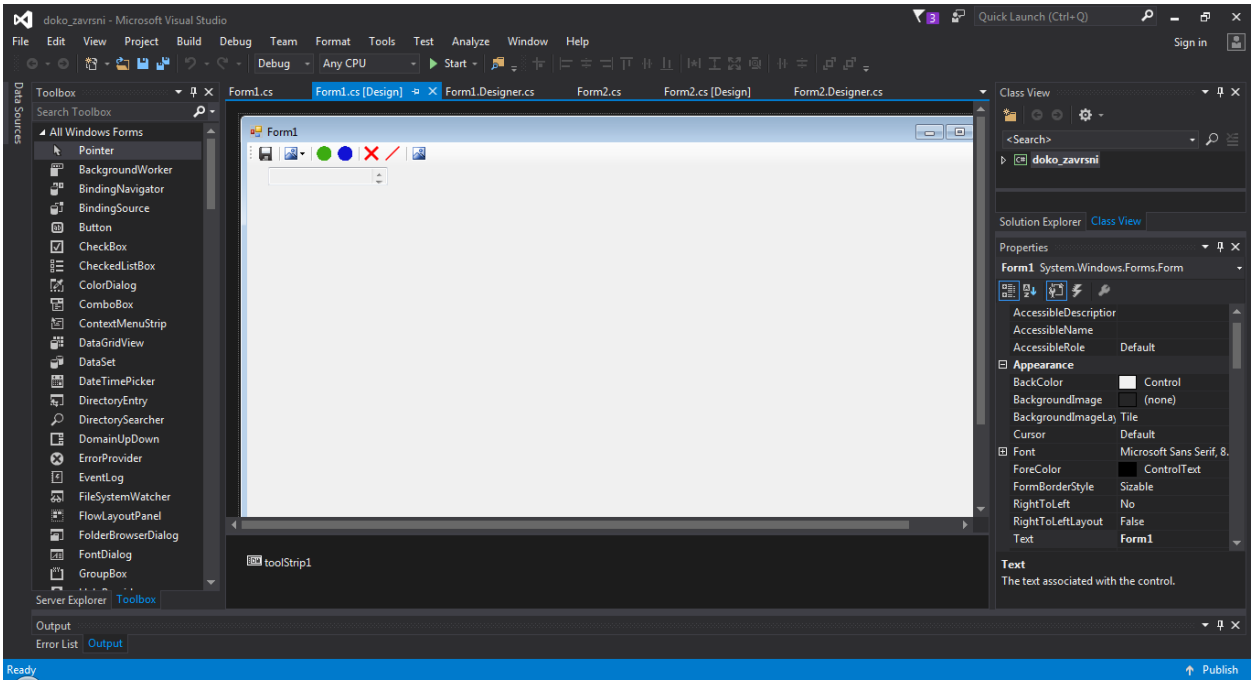


Slika 2.2 Prikaz mogućih predložaka C# aplikacije

3. APLIKACIJA

3.1 Aplikacija

Pomoću navedenih alata, za izradu ove aplikacije prvo je potrebno urediti vizualni izgled aplikacije koja mora biti jednostavna za korištenje. C# aplikacija pri svome pokretanju automatski izbacuje prazni predložak Windows forme na koju se mogu dodavati različiti textboxovi, labele, buttoni i slično. Na slici 3.1 može se vidjeti izgled aplikacije i sa lijeve strane izbornik Toolbox sa puno opcija za izradu aplikacije.

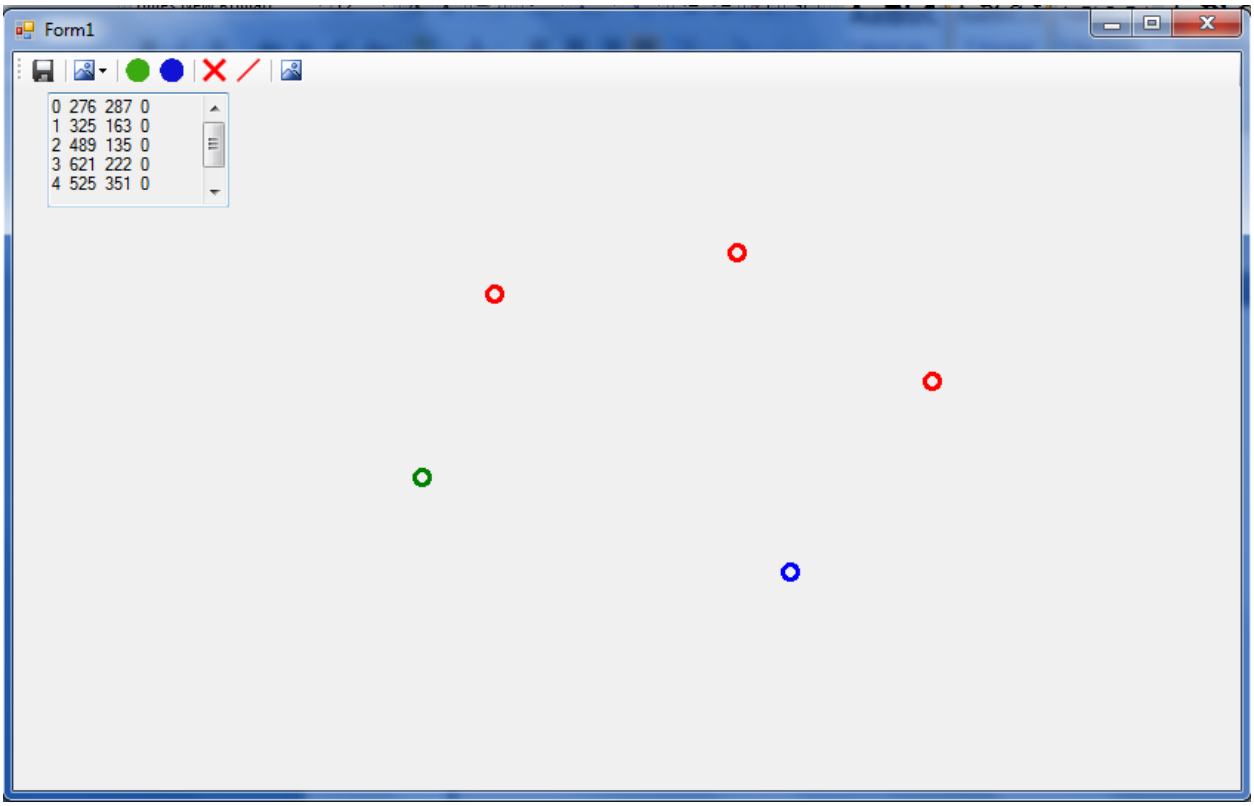


Slika 3.1 Izgled aplikacije i izbornik Toolbox

Iako grafički prikaz aplikacije na prvi pogled izgleda vrlo jednostavno, tu ima još dosta mogućnosti i pozadinskog koda koji omogućuje nesmetani rad aplikacije.

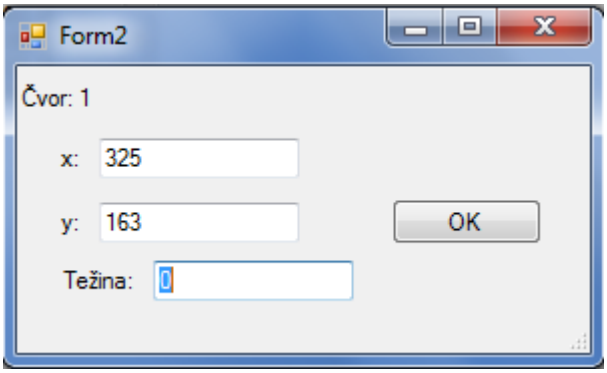
Kada se pokrene aplikacija pritiskom na Start ili F5, otvara se aplikacija i tada je moguće u nju dodavati čvorove. U dogovoru sa mentorom pritiskom lijevog klika miša dodaju se čvorovi, dok se pritiskom desnog klika miša na već dodani čvor taj isti čvor briše. Za dodavanje čvorova prvo je bilo potrebno definirati polje čvorova. Maximalni broj čvorova koji se mogu dodati u polje je 50(0-49). Pritiskom lijevog klika miša na prazni prostor u aplikaciji dodaje se čvor i otvara se popup prozor sa mogućnostima postavljanja koordinata x,y te dodavanje određene težine čvoru i nakon toga čvor se sprema u polje čvorova. Za vizualni prikaz čvora koristi se funkcija DrawCircle čiji su atributi koordinate x i y, te širina i visina. Početni čvor u polju prikazan je

zelenom bojom dok je krajnji čvor u polju plave boje. Svi čvorovi između početnog i krajnjeg čvora su crvene boje.



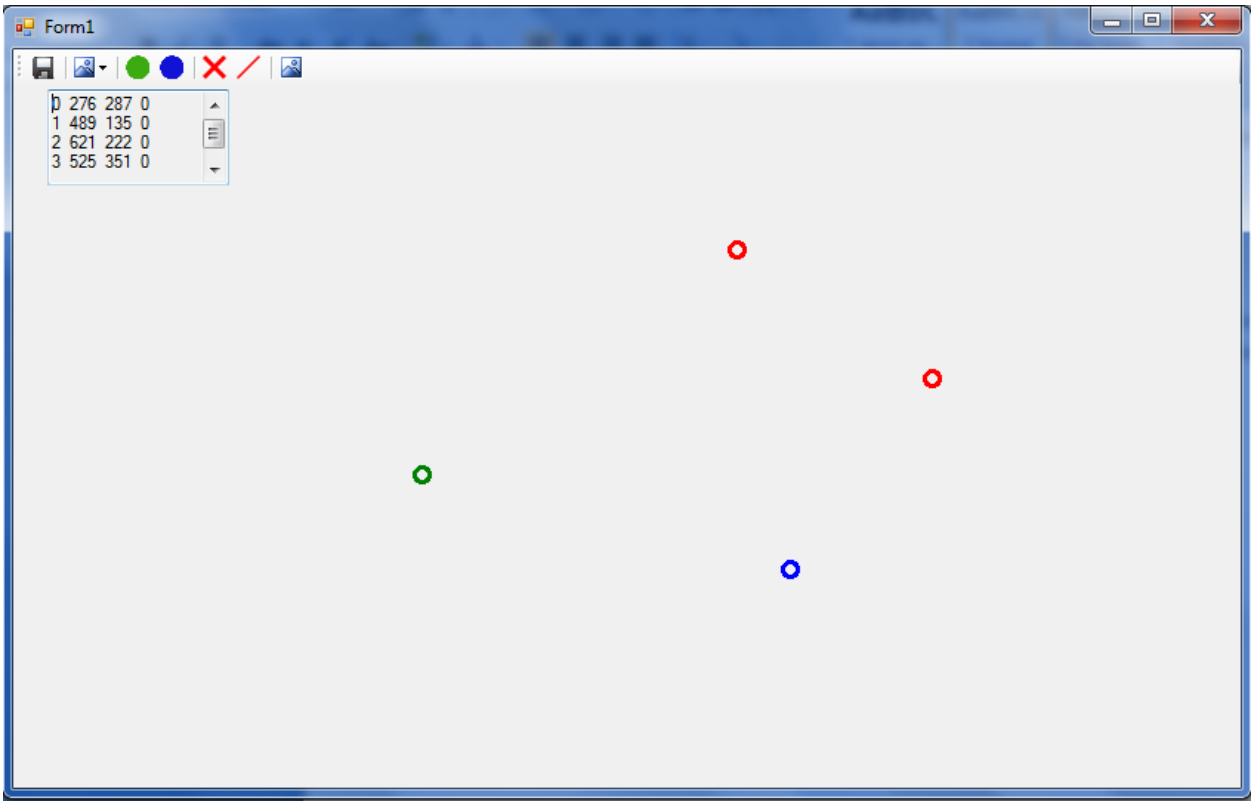
Slika 3.2 Vizualni prikaz pet čvorova dodanih lijevim klikom miša

Kada se dodaju čvorovi u texboxu u lijevom kutu pojavljuje se ispis svih čvorova. Svaki čvor ima svoj index u polju(0-49), koordinatu x, koordinatu y i težinu. Prilikom dodavanja novog čvora ili lijevim klikom na već postojeći čvor, otvara se skočni(pop-up) prozor gdje se mogu unijeti nove, odnosno promijeniti koordinate već prije dodanog čvora u polje i može se dodijeliti težina čvoru.



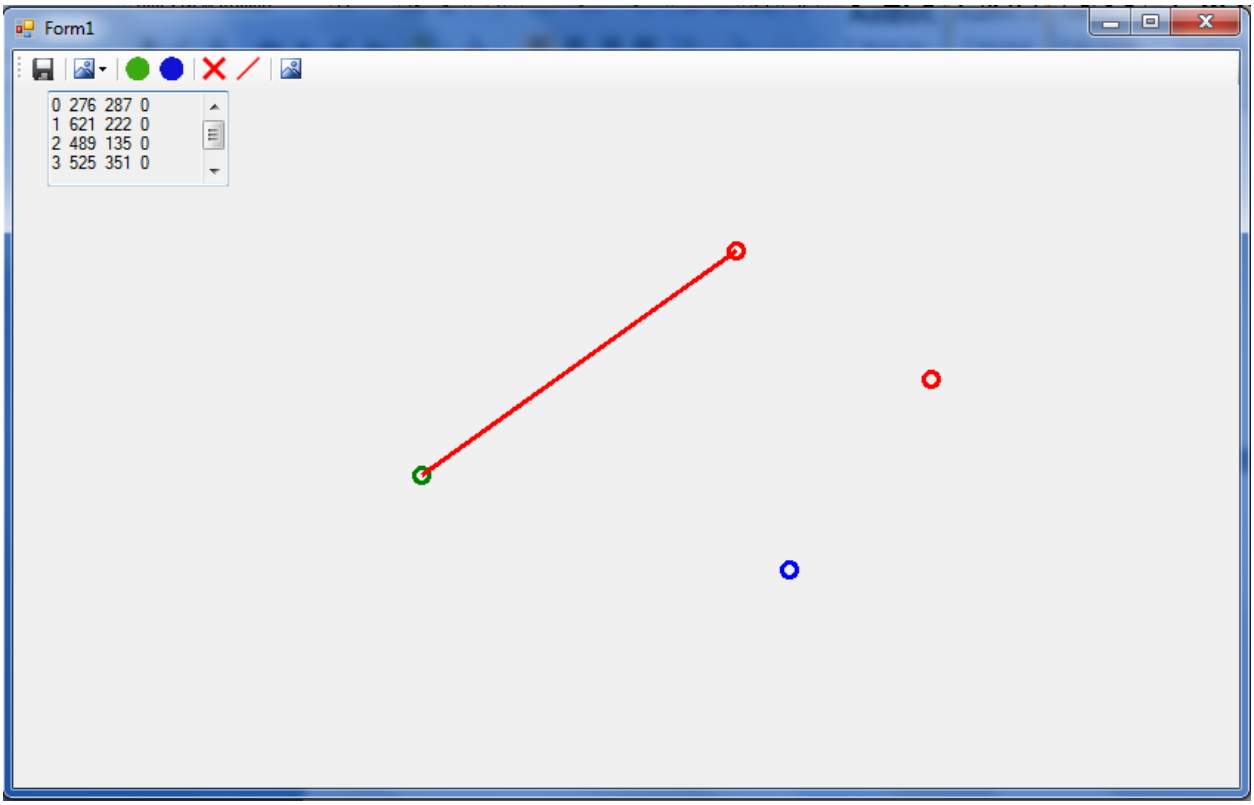
Slika 3.3 Uređivanje čvora sa indexom broja 1 u polju

Kliknutom čvoru mogu se promijeniti koordinate x i y i dodijeliti težina. Pritiskom na button OK primjenjuju se promijene(ako ih je bilo) i zatvara se Form2. Za brisanje čvorova potrebno je kliknuti desnim klikom miša na čvor koji želimo ukloniti. Prvo se provodi ispitivanje da li je desnim klikom kliknut čvor sa funkcijom checkNode. Ako se klikne desnim klikom na prazninu neće se ništa dogoditi, no ako kliknemo na čvor tada se on briše iz grafičkog prikaza, ali i iz polja jer se taj čvor više ne nalazi u polju. Kada se čvor izbriše iz polja, svim čvorovima sa indeksom većim od njega(odnosno sljedeći čvorovi u nizu) smanjuje se indeks u polju za 1.

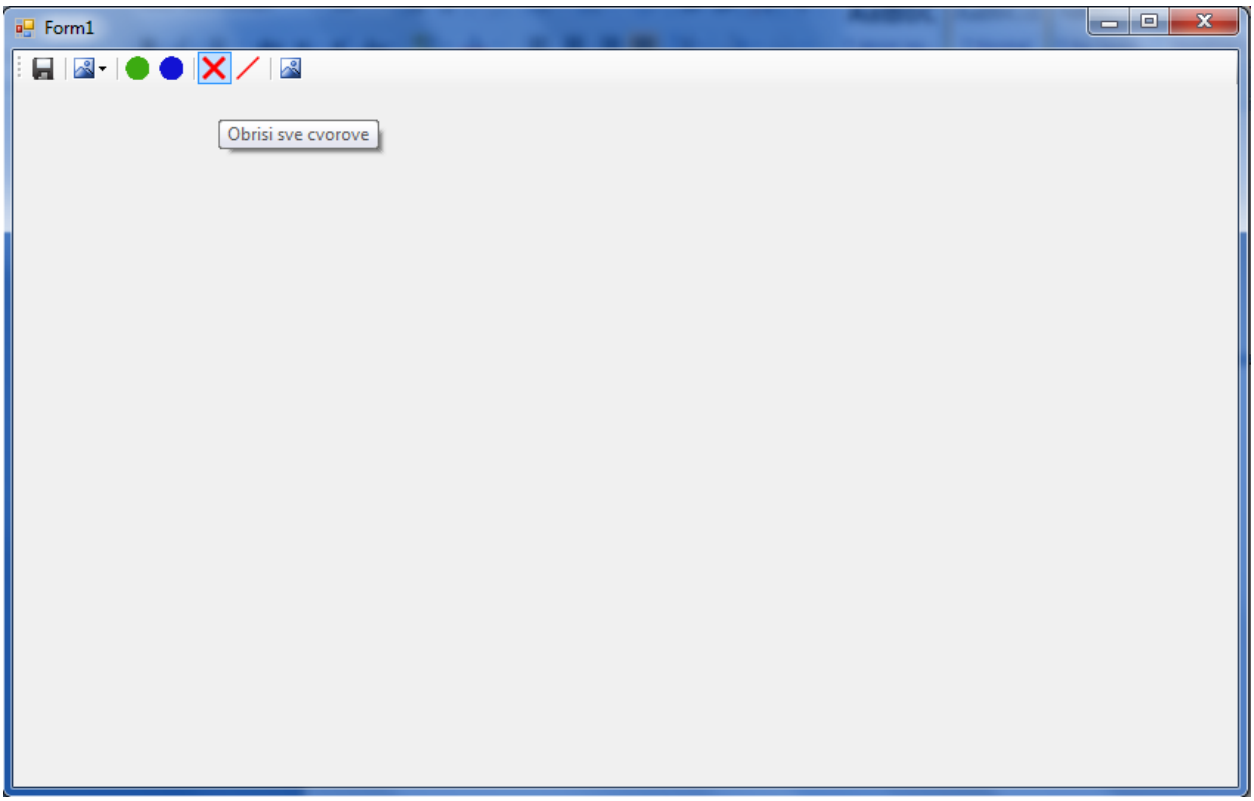


Slika 3.4 Brisanje čvora 1 iz polja

Ako se obriše početni čvor(zeleni), potrebno je pritiskom na zeleni gumb odabrati novi početni čvor. Isto se odnosi i na kranji čvor(plavi), ako se obriše kranji čvor. Pritiskom na zeleni/plavi gumb može se promijeniti početni/krajnji čvor u polju.Desno od plavog gumba nalaze se još dva gumba za uređivanje čvorova. Pritiskom na crveni gumb sa znakom X(iks) brišu se svi dodani čvorovi u polju čvorova. Pored crvenog znaka X nalazi se gumb koji omogućuje povezivanje dva čvora crvenom linijom tako što prvo odaberemo početni čvor, a zatim kranji čvor i ta dva čvora su povezana.



Slika 3.5 Povezivanje dva čvora

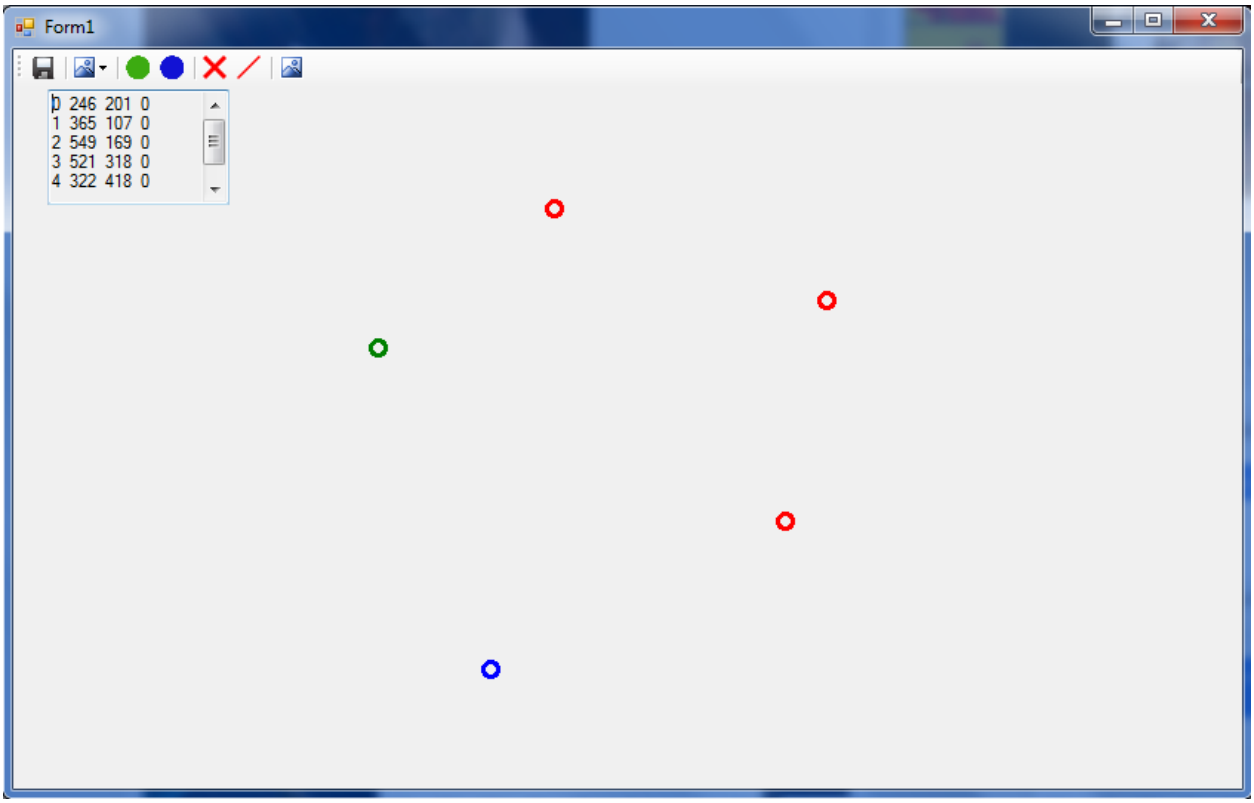


Slika 3.6 Brisanje svih čvorova

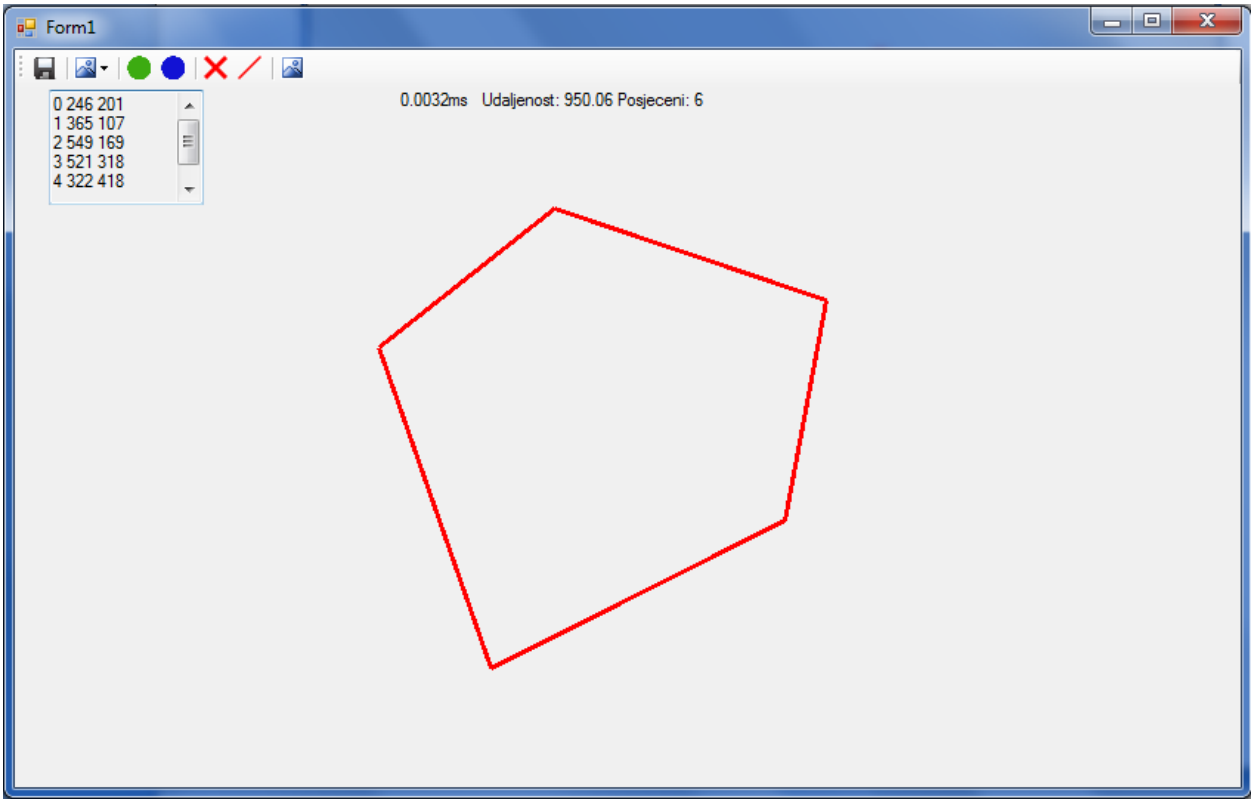
Nakon što se urede čvorovi po želji možemo odabrati neku od metoda pretraživanja koje su dodane u radu. U ovome radu mogu se odabrati tri metode pretraživanja: Greedy(odnosno nearest neighbour) algoritam, random algoritam i Dijkstrin algoritam.

3.2 Nearest neighbour i random algoritam

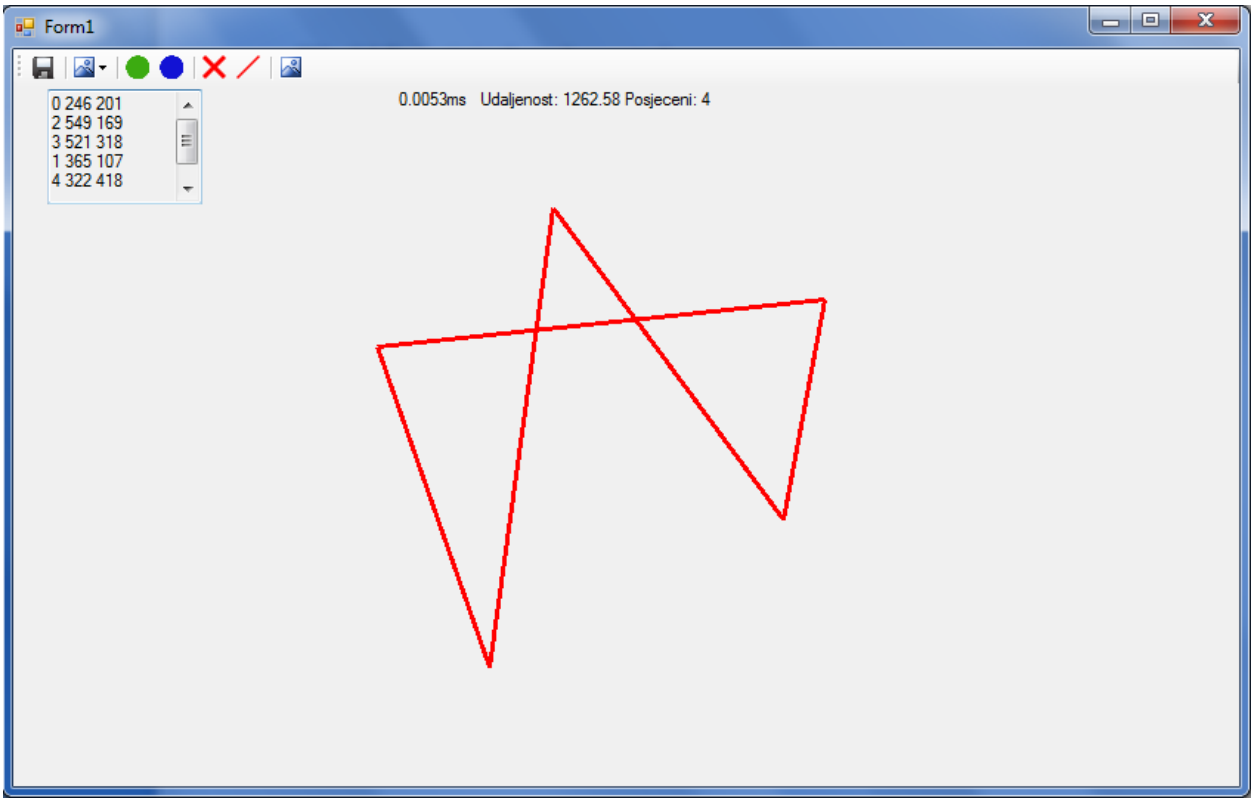
Nearest neighbour i random algoritam su metode koje ne koriste težine prilikom pretraživanja puta od početnog do kranjeg čvora(vrijednost težine je 0) i ne treba povezivati čvorove međusobno jer kad se odabere jedna od te dvije metode one same povežu čvorove crvenim linijama. Nearest neighbour metoda pretražuje najkraću udaljenost između početnog i njemu sljedećeg najbližeg čvora. Nakon što se pronađe najbliži čvor, tada pronađeni čvor traži njemu najbliži čvor i tako sve do zadnjega čvora. Na kraju metode povežu se krajnji i početni čvor. Najkraća udaljenost računa se kao drugi korijen zbroja između razlika koordinata na kvadrat, odnosno minDist=sqrt($(x_2-x_1)^2+(y_2-y_1)^2$). Složenost nearest neighbour algoritma je $O(n^2)$ gdje n predstavlja broj čvorova. Što je više čvorova to je metoda sve složenija, odnosno njena složenost raste eksponencijalno. U aplikaciji je moguće dodati najviše 50 čvorova što znači da je maksimalna složenost $O(50^2)$. Kod random metode postupak je jednostavniji jer odabirom te metode ne računa se najkraća udaljenost između čvorova već se oni povezuju na slučajan odabir i kada se povezuju slučajno rješenje je uvijek različito, dok kod nearest neighbour metode nije tako, rješenje je samo jedno. Složenost random metode je O(n) gdje n također predstavlja broj čvorova. Što je više čvorova to je metoda složenija, ali složenost kod ove metode raste linearno za razliku od nearest neighbour metode gdje složenost raste eksponencijalno. Ova metoda je manje složenija, ali je nepreciznija od nearest neighbour metode kada se traži najmanja udaljenost od početnog do krajnjeg čvora. Nakon nasumičnog odabira čvorova čija je težina nula možemo primijeniti nearest neighbour ili random metodu.



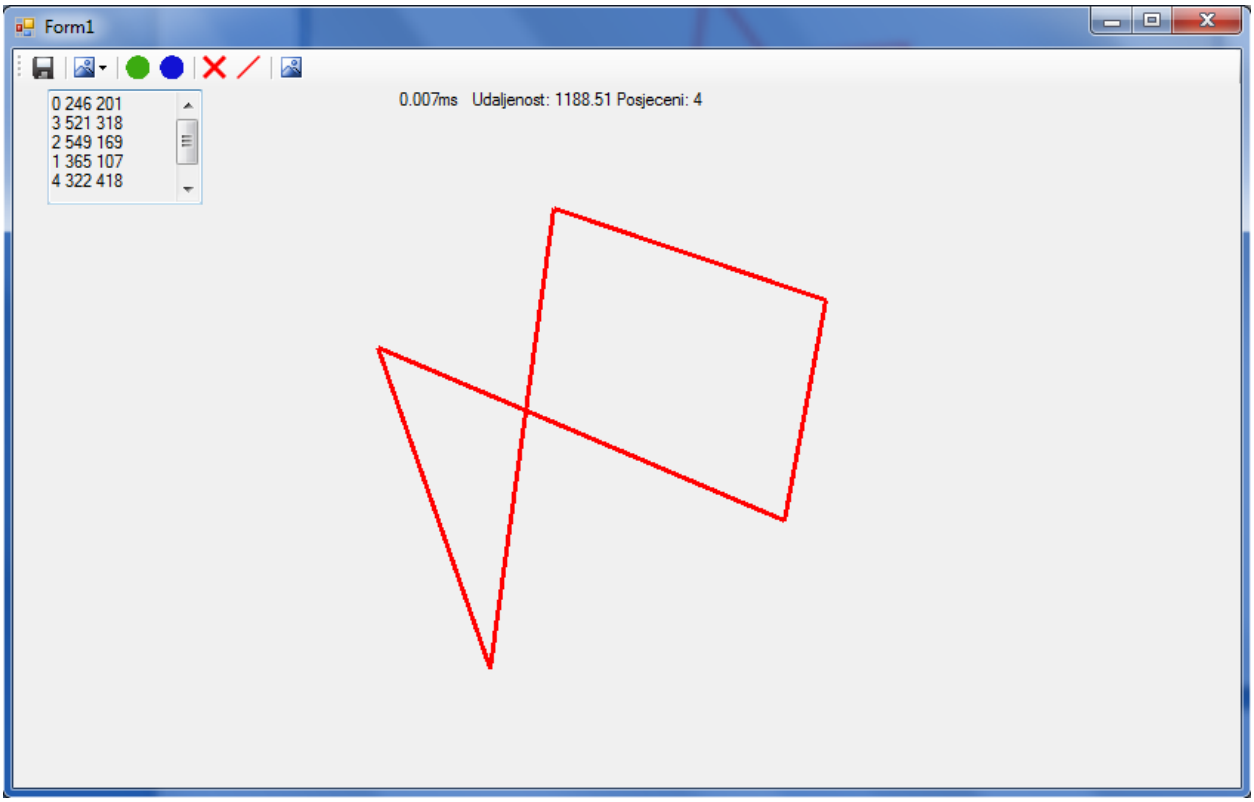
Slika 3.7 Nasumično odabrani čvorovi



Slika 3.8 Nearest neighbour metoda



Slika 3.9 Jedno od rješenja random metode

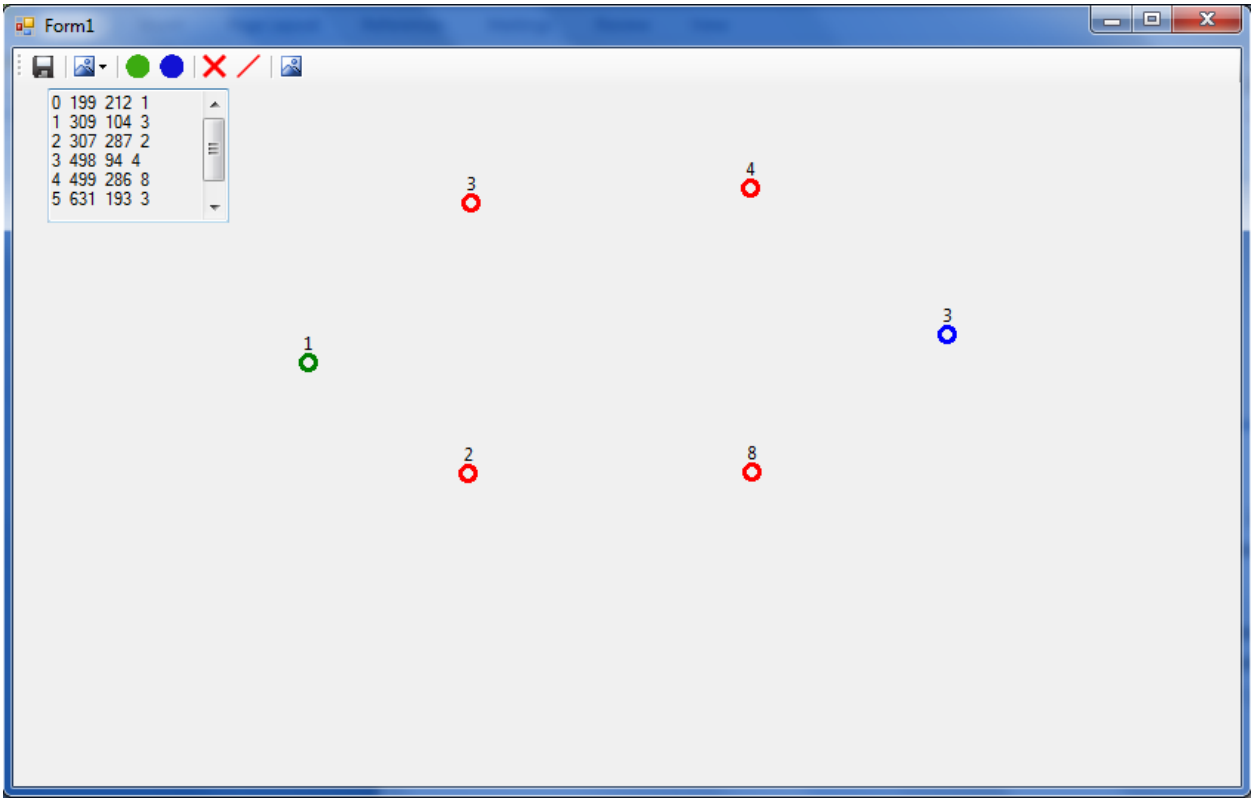


Slika 3.10 Jedno od rješenja random metode

Nakon odabira metode za pretraživanje, čvorovi se povežu i ispisuju se rezultati pretraživanja. Ispisuje se vrijeme pretraživanja u milisekundama(ms), ukupna udaljenost između početnog i krajnjeg čvora, te broj posjećenih čvorova u polju čvorova.

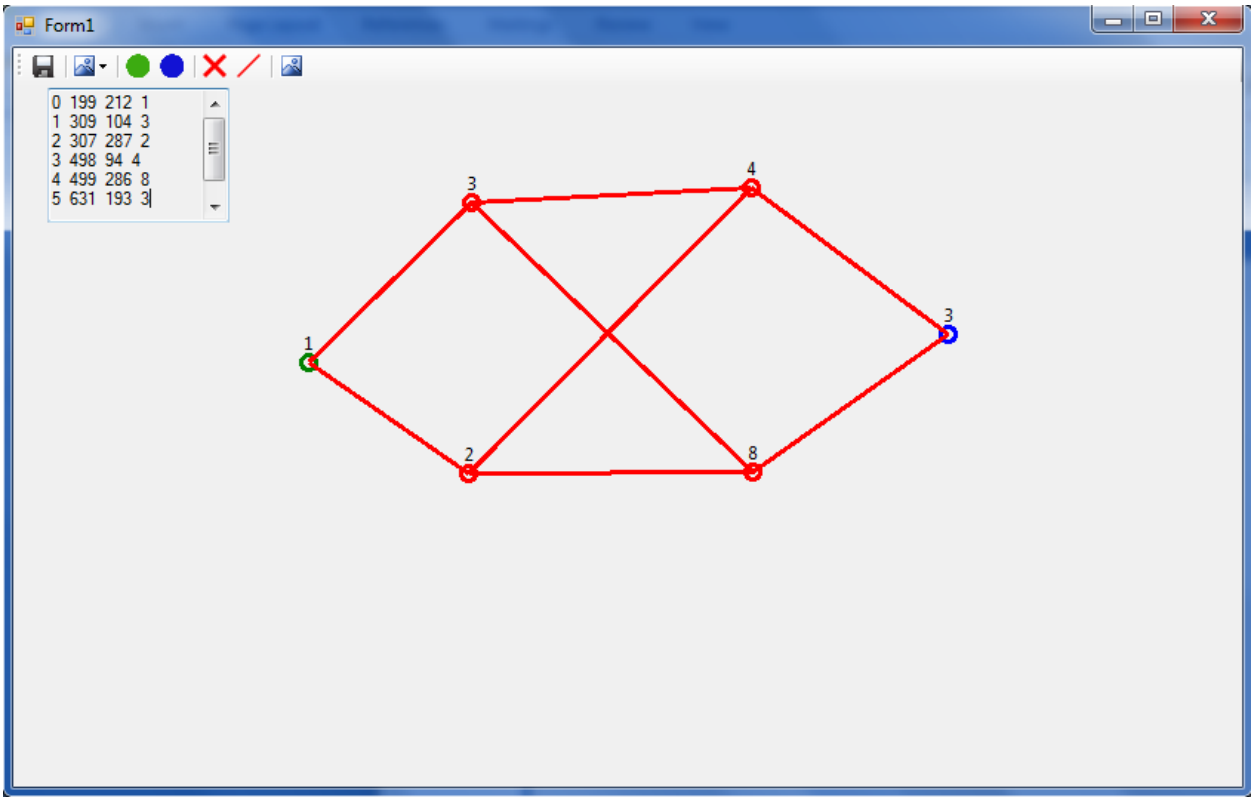
3.3 Dijkstrin algoritam

Dijkstrin algoritam radi na drugačijem principu od ostale dvije navedene metode. Prilikom dodavanja čvorova ili već dodanim čvorovima potrebno je dodijeliti određenu težinu. Nakon dodavanja težine potrebno je povezati čvorove i poslije toga se može primijeniti Dijkstrin algoritam. Lijevim klikom na već postojeći čvor otvara se skočni(pop-up) prozor gdje se može dodijeliti težina čvoru. Čvorovi se povezuju tako da se prvo klikne na prvi čvor, a zatim na drugi čvor koji se želi povezati i tada su ta dva čvora povezana.



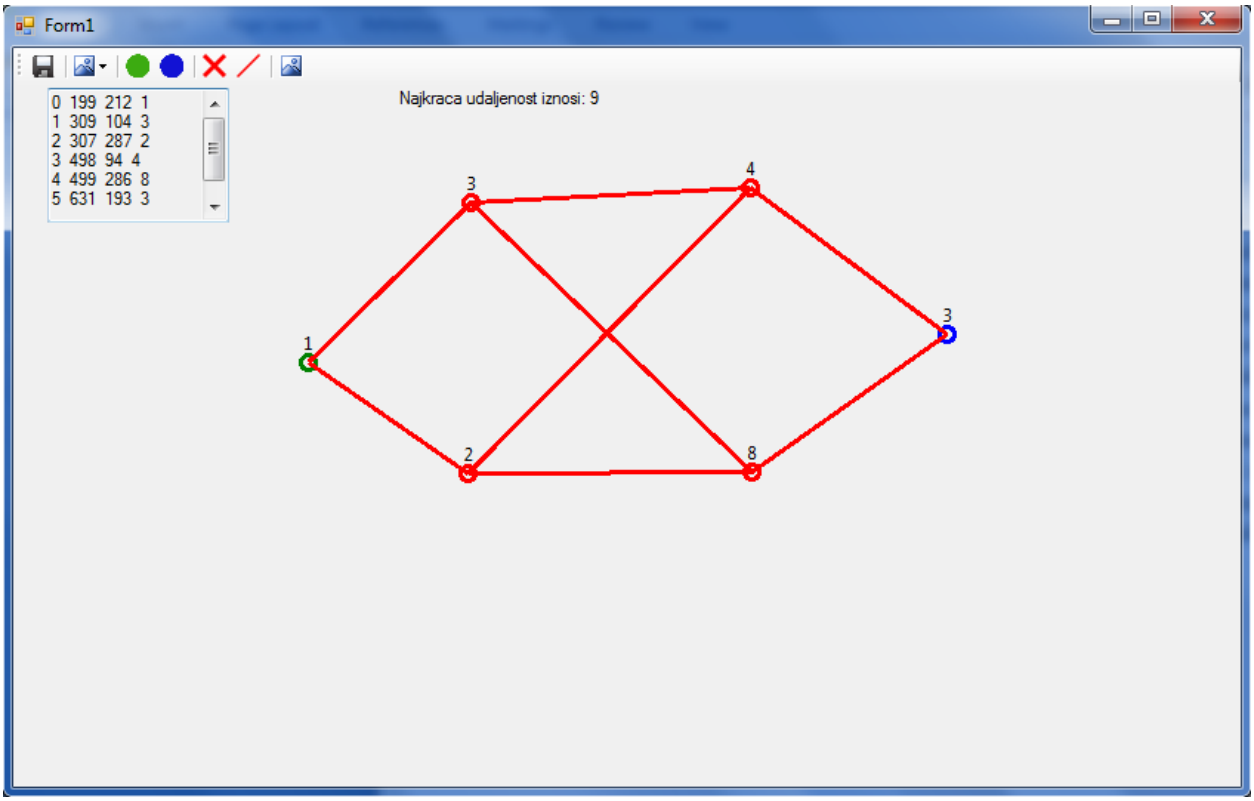
Slika 3.11 Šest čvorova sa zadanim težinama

Nakon dodavanja težine čvorovima, može ih se povezati po želji.



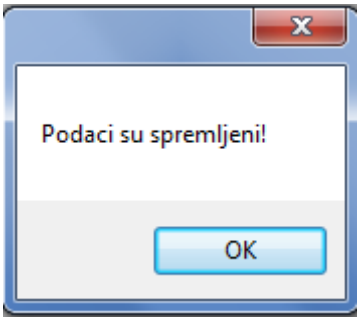
Slika 3.12 Šest čvorova povezani po želji.

Pritiskom na gumb s Dijkstrin metodom izračunava se najkraća udaljenost između povezanih čvorova. Dijkstrin algoritam radi na principu da početni čvor traži idući povezani čvor sa najmanjom težinom. U ovome primjeru početni čvor povezan je sa čvorovima čija su težina dva i tri, čvor sa težinom dva ima manju težinu i on postaje novi čvor od kojega se dalje gleda povezanost sa čvorom sa najmanjom težinom. Čvor sa težinom dva je povezan sa čvorovima težine četiri i osam, čvor sa težinom četiri postaje novi čvor od kojega se dalje gleda povezanost sa čvorom sa najmanjom težinom. Čvor sa težinom četiri je povezan sa čvorom težine tri, tri(krajnji, odredišni čvor) i dva. Ovdje stajemo sa algoritmom jer smo pronašli krajnji čvor. Složenost ovog algoritma je O(m*log(n)), gdje m predstavlja broj veza između čvorova, a n predstavlja broj čvorova. Ovo je vrlo precizna metoda za određivanja najmanje udaljenosti od početnog do krajnjeg čvora, ali ona zahtjeva puno više podataka i veza od ostale dvije metode. Cilj Dijkstrin algoritma je proći kroz cijelu mrežu povezanih čvorova i pronaći najmanju udaljenost po težini od početnog do krajnjeg čvora. Najkraća udaljenost po težini je ukupno 9(2+4+3). Neka od drugih rješenja bi bila na primjer: 10(3+4+3), 14(3+8+3), 20(3+8+2+4+3), ali to nisu optimalna rješenja za ovaj algoritam jer se traži najmanja udaljenost po težini.

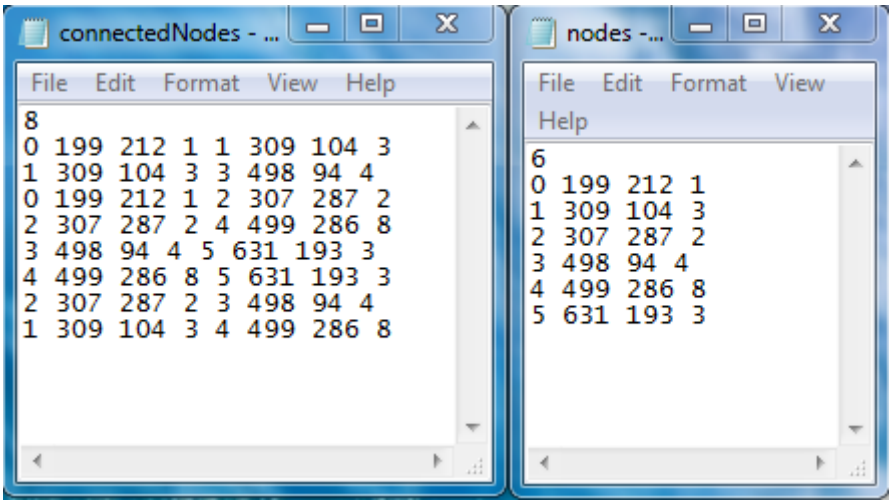


Slika 3.13 Ispis rezultata pretrage Dijkstrin algoritma za povezane čvorove

Kada je korisnik gotov sa radom i želi spremiti podatke, točnije čvorove sa kojima je radio i na koje su primijenjene metode, pritiskom na gumb Save spremaju se podatci u dva .txt dokumenta. Dokumenti se spremaju u direktorij gdje je spremljena aplikacija i imaju nazive connectedNodes i nodes. Datoteka nodes ima spremljene u sebi posljednje pohranjene čvorove zajedno sa njihovim koordinatama i težinama. Datoteka connectedNodes sadrži u sebi također čvorove zajedno sa koordinatama i težinama, ali su ti čvorovi poredani u datoteci tako da predstavljaju međusobno povezane čvorove koji se koriste za Dijkstrin algoritam.



Slika 3.14 Skočni prozor kada se pohrane podaci



Slika 3.15 Datoteke u koje su spremljene veze i čvorovi

Prvi broj u connectedNodes datoteci predstavlja ukupan broj veza, dok u drugoj datoteci(nodes) predstavlja ukupan broj čvorova.

4. ZAKLJUČAK

U ovome radu napravljena je C# aplikacija za pretraživanje grafova. Omogućeno je dodavanje i uređivanje čvorova i nad tim čvorovima se mogu primijeniti metode pretraživanja. Metode primijenjene u radu su random, nearest neighbour i Dijkstrin algoritam. Te metode imaju različite složenosti, ali imaju sličnu primjenu. Složenost metode random je najmanja i ona iznosi O(n), gdje n predstavlja broj čvorova, ali ujedno ta metoda je i najmanje precizna što se tiče određivanja puta od početne do kranje točke. Nearest neighbour metoda ima složenost $O(n^2)$ gdje n također predstavlja broj čvorova i ona je preciznija od random metode. Složenost random metode raste linearno povećanjem broja čvorova, dok složenost nearest neighbour metode raste eksponencijalno. Dijkstrin algoritam je metoda koja ima složenost O(m*log(n)) gdje m predstavlja broj linija(poveznica) između čvorova, a n predstavlja broj čvorova. Dijkstrin metoda je najpreciznija od metoda navedenih u ovome radu. Metode navedene u radu mogu se primijeniti u bilo kojem primjeru gdje se treba odrediti najbolji, odnosno optimalni put od ishodišne do odredišne točke. Za primjer se može uzeti dva grada kao početna i krajnja točka, a ceste između predstavljaju različite udaljenosti(putanje) do tih gradova.

LITERATURA

[1] Crtanje elipse/kruga u c#, http://stackoverflow.com/questions/29055863/draw-ellipse-with-painteventargs (lipanj, 2016.)

[2] Nearest neighbour algoritam, https://www.seas.gwu.edu/~simhaweb/champalg/tsp/tsp.html (lipanj, 2016.)

[3] Dijkstrin algoritam, https://www.youtube.com/watch?v=WN3Rb9wVYDY (rujan, 2016.)

[4] Microsoft Visual Studio, https://bs.wikipedia.org/wiki/Microsoft_Visual_Studio (lipanj, 2016.)

[5] C#, https://en.wikipedia.org/wiki/C_Sharp_(programming_language) (lipanj, 2016)

SAŽETAK

Naslov: C# aplikacija za pretraživanje grafova.

C# aplikacija za pretraživanje grafova napravljena je u programu Microsoft Visual Studio 2015. U aplikaciji omogućeno je dodavanje čvorova na koje se mogu primijeniti metode pretraživanja. Čvorovi se mogu dodavati lijevim, a brisati desnim klikom. Prilikom dodavanja ili klikom na već postojeći čvor otvara se skočni(pop-up) prozor u kojemu se mogu urediti koordinate x,y i težina čvora. Početni čvor prikazan je zelenom bojom, krajnji čvor plavom, a svi čvorovi između su crvene boje. Pritiskom na gumb sa crvenim X znakom brišu se svi čvorovi, a sa gumbom kose crvene linije mogu se povezivati čvorovi međusobno. Nakon dodavanja čvorova bez težina mogu se primijeniti nearest neighbour i random metoda pretraživanja. Nakon odabira jedne od te dvije metode, prikazuje se ispis vremena pretraživanja, udaljenost od početnog do krajnjeg čvora i broj posjećenih čvorova. Za primjenu Dijkstrin algoritma potrebno je prvo čvorovima dodijeliti težine(težinske vrijednosti) i nakon toga povezati ih međusobno. Kada se primijeni Dijkstrin algoritam ispisuje se najkraća udaljenost od početnog do krajnjeg čvora preko povezanih čvorova. Kada je korisnik gotov sa radom i želi spremiti podatke, pritiskom na gumb Save u direktoriju gdje je spremljena aplikacija spremaju se dva .txt dokumenta od kojih prvi sadrži čvorove sa njihovim koordinatama i težinama(dokument nodes), a drugi način na koji su povezani ti čvorovi(dokument connectedNodes).

Ključne riječi: C#, čvor, nearest neighbour, random, Dijkstrin.

ABSTRACT

Title: C# application for graph search.

C# application was made using Microsoft Visual Studio 2015. In the application users can add nodes on which methods for searching can be applied. Nodes can be added with left mouse click, and they can be removed by right mouse click. When a node is added or an existing node is clicked, a pop-up window appears in which x and y coordinates and the weight of the node can be edited. The beginning node is coloured green, the end node blue, and every node inbetween is coloured red. By clicking on the button with the red X, all nodes are deleted, and by clicking on the button with a red askew line, all nodes can be connected. After adding weight-less nodes, nearest neighbour and random search methods can be applied. After choosing one of those two methods, time of the search, distance from the beginning and end node and the number of visited nodes is shown. To apply Dijkstra's algorithm, nodes first need to be assigned weight and after that, the nodes need to be connected. When Dijkstra's algorithm is applied, shortest distance from the beginning to the end node is shown. When the user is done working and wants to save the data, clicking on the button Save, two .txt files are saved. The first contains nodes with their coordinates and weights and the other one contains the way those nodes are connected.

Keywords: node, nearest neighbour, random, Dijkstra's, C#

ŽIVOTOPIS

Petar Doko rođen je 12.2.1995. godine u Đakovu. Živi na adresi Biskupa Ćirila Kosa 3 u Đakovu. Nakon završetka osnovne škole „Vladimir Nazor“ 2009. godine u Đakovu, upisuje Opću gimnaziju u Đakovu te maturira 2013.g. Iste godine upisuje sveučilišni studij na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija, smjer računarstvo. Vrlo dobro se služi engleskim jezikom i posjeduje osnovna znanja za korištenje Microsoft Office alatima. Posjeduje srednju razinu znanja u korištenju programskih jezika C, C++ i C#

PRILOG

U prilogu se nalazi CD sa pdf dokumentom završnog rada i zip datoteka sa aplikacijom napravljenom u Microsoft Visual Studio 2015.