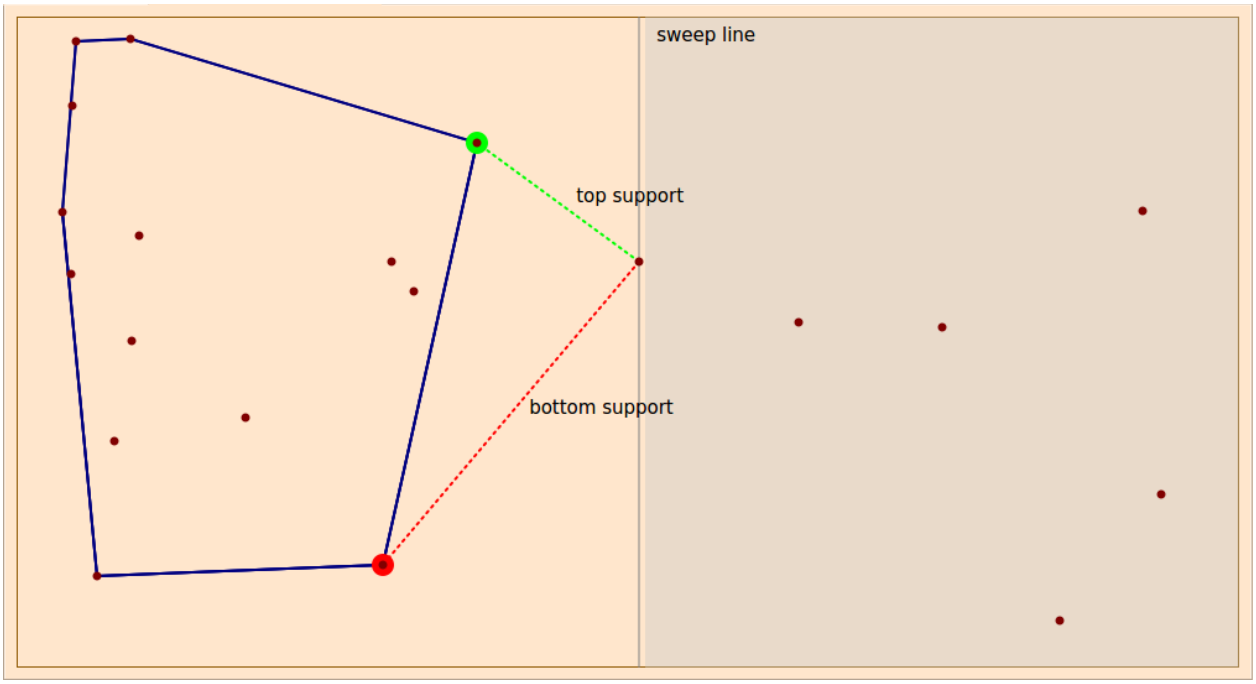
Geometrijski algoritmi @ MATF

Inkrementalni i QuickHull - algoritmi za izračunavanje konveksnog omotača skupa tačaka u ravni

Đorđe Milićević



Opis problema

Skup tačaka X (ravni ili prostora) je konveksan ako za svake dve tačke A i B skupa X svaka tačka duži AB pripada skupu X. Poligon je konveksan ako su za svaku njegovu stranicu sve njegove tačke sa iste njene strane, tj. ako su mu svi unutrašnji uglovi manji od opruženog. Konveksni omotač skupa tačaka je najmanji konveksan skup tačaka koji sadrži X. Za konačan skup tačaka u ravni, konveksni omotač je

(konveksni) poligon. Za fiksiran skup tačaka, konveksni omotač je određen jednoznačno.

Konveksni omotač ima razne primene, a neke od njih su: u okviru mnogih algoritama računarske geometrije, obradi slika, prepoznavanju oblika, statistici i geografskim infomacionim sistemima.

Ulaz: *skup od n tačaka u ravni*

Izlaz: *skup tačaka koje predstavljaju konveksan omotač*

Naivno rešenje problema

Za naivno rešenje problema iskorišćen je algoritam čija je složenost u najgorem slučaju $O(n^3)$. Iako je algoritam korektan, zbog zaokruživanja u aritmetici sa pokrenim zarezom, njegova implementacija neće uvek dati ispravan rezultat (u slučaju kada su neke tri tačke skoro kolinearne). Naivni algoritam možemo predstaviti sledećim pseudokodom:

```
Ulaz: skup S koji sadrži n tačaka u ravni (n > 2)

Izlaz: tačke koje čine konveksni omotač početnog skupa tačaka u ravni


Naivni(S):

	za sve parove različitih tačaka P i Q iz skupa S:

		valid = true;

		za svaku tačku R iz S koja je različita od P i Q:

			if (trojka PQR je negativno orijentisana)

				valid = false;

		if (valid == true)

			dodaj tačke P i Q u konveksni omotač;
```

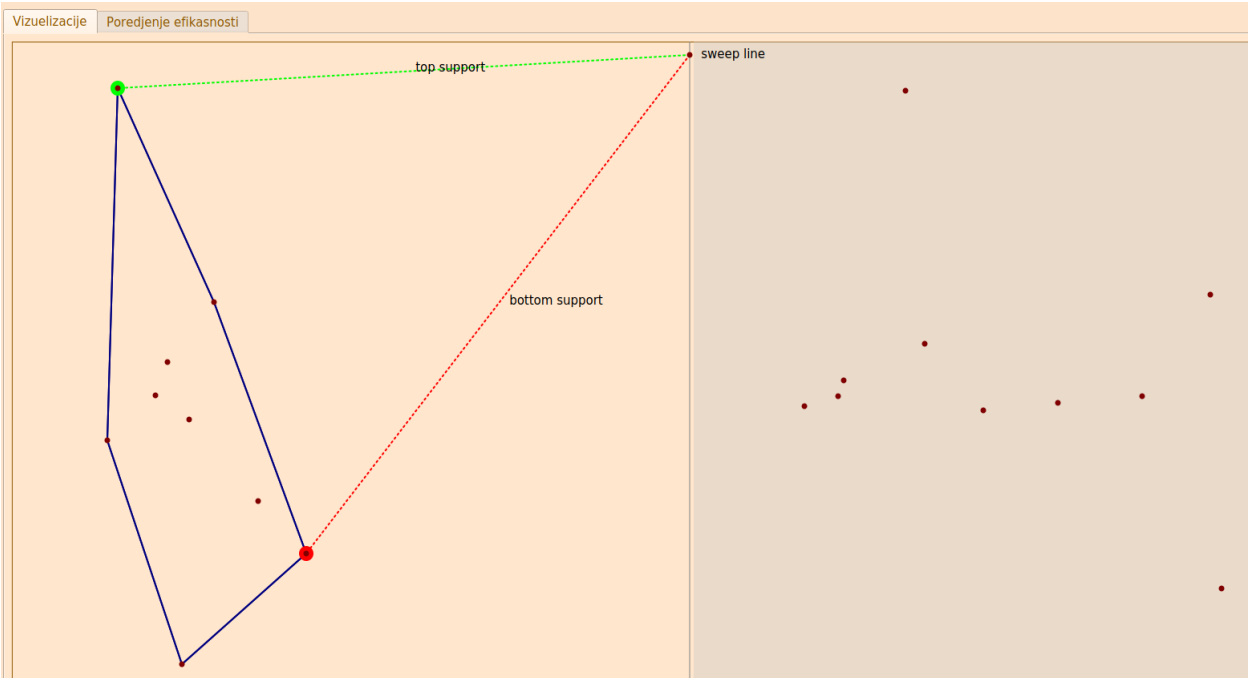
Inkrementalni algoritam

Inkrementalni algoritam za konstruisanje konveksnog omotača zasnovan je na metodi brišuće prave. Konstrukcija konveksnog omotača vrši se pomeranjem vertikalne brišuće prave sleva nadesno. Tačke događaja su tačke iz ulaznog skupa tačaka. Kao struktura podataka, koristi se dvostruko povezana ciklična lista u kojoj svaki čvor ima pokazivač na sledeći i prethodni čvor. Složenost algoritma u najgorem slučaju je O(n*log(n)). Algoritam se sastoji iz tri važne celine:

- sortiranje tačaka po x koordinati (ako neke dve tačke imaju istu x koordinatu, porede se po y koordinati)
- konstruisanje pozitivno orijentisanog trougla od prve tri tačke (usmeravanjem određenih pokazivača ka određenim tačkama)
- konstruisanje ostatka konveksnog omotača inkrementalnim dodavanjem preostalih tačaka

Glavni zadatak je dodavanje nove tačke u postojeći konveksni omotač. Akcija dodavanja može prouzrokovati modifikaciju prethodno konstruisanog konveknog omotača. Neka je P tačka koju dodajemo u postojeći konveksni omotač. Potrebno je pronaći dve potporne prave koje prolaze kroz tačku P i podupiru postojeći konveksni omotač sa donje i gornje strane. Potporne prave će činiti stranice novog omotača. Sve tačke koje su činile granicu prethodnog konveksnog omotača, a sada se nalaze u njegovoj unutrašnjosti, nestaju iz novog konveksnog omotača. Ovaj korak možemo predstaviti sledećim pseudokodom:

```
topP = bottomP = lastP;

// pronalaženje tačke na trenutnom omotaču kroz koju će proći gornja prava

while (on_right_side(currentP, topP, topP->next))

	topP = topP->next;

// pronalaženje tačke na trenutnom omotaču kroz koju će proći donja prava

while (on_left_side(currentP, bottomP, bottomP->prev))

	bottomP = bottomP->prev;

// ažuriranje liste

currentP->next = topP;

currentP->prev = bottomP;

topP->prev = currentP;

bottomP->next = currentP;
```



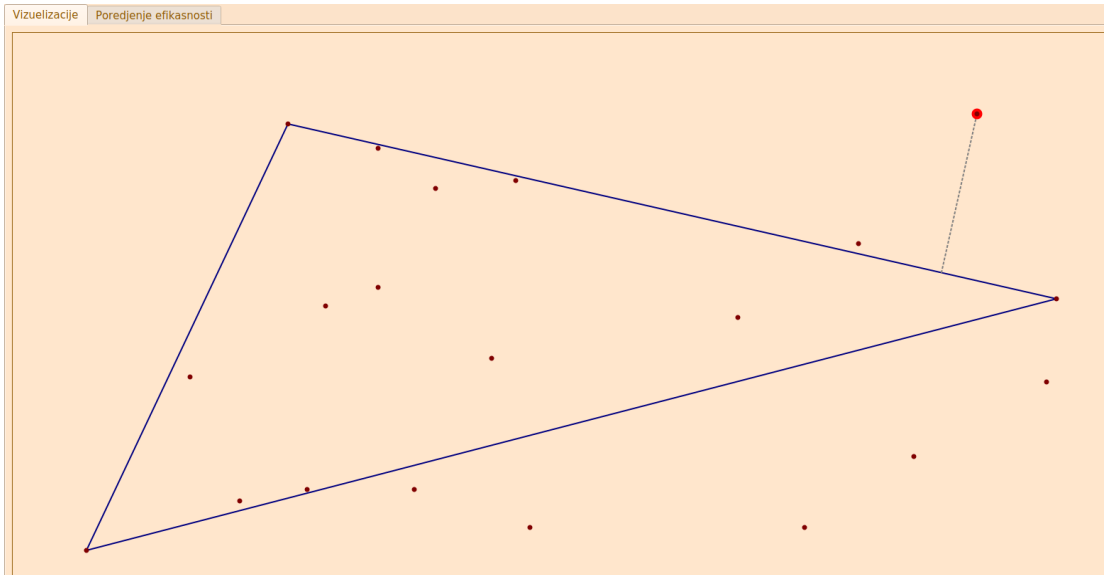
Inkrementalni algoritam

Na slici možemo videti dve potporne prave. Gornja prava je zelene boje, dok je donja obojena crvenom bojom.

QuickHull algoritam

Algoritam QuickHull predstavlja jedan od načina za izračunavanje konveksnog omotača konačnog skupa tačaka u ravni. Koristi se strategija “podeli pa vladaj”. Složenost algoritma u prosečnom slučaju je O(n*log(n)), dok je složenost u najgorem slučaju $O(n^2)$. Algoritam možemo predstaviti sledećim pseudokodom:

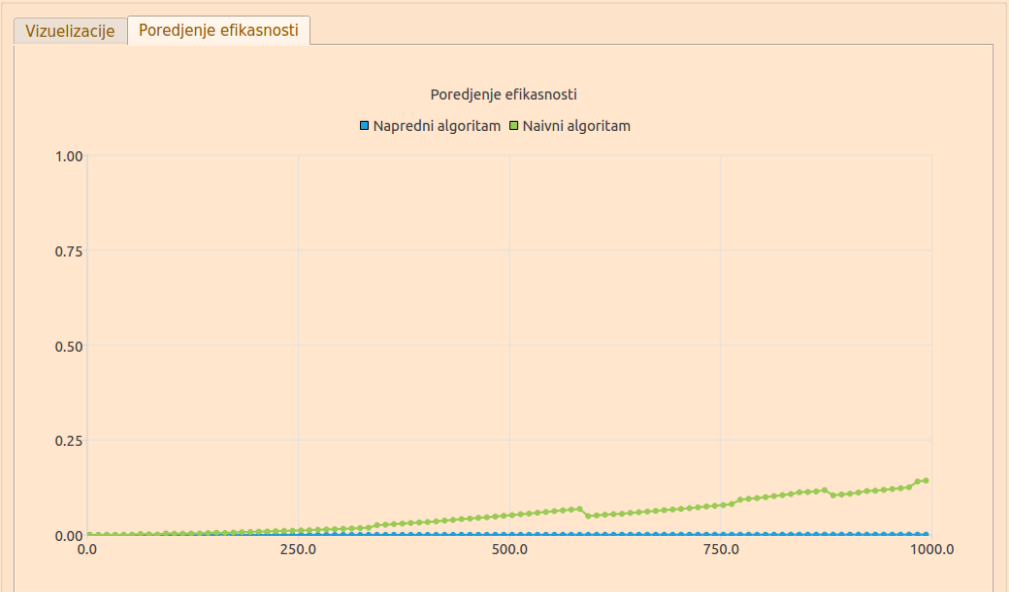
```
Ulaz: skup S koji sadrži n tačaka u ravni (n > 2)

Izlaz: tačke koje čine konveksni omotač početnog skupa tačaka u ravni

QuickHull(S):
        pronaći tačku A sa najmanjom x koordinatom;
        pronaći tačku B sa najvećom x koordinatom;
        tačke A i B dodati u konveksni omotač;
        // duž AB deli dati skup tačaka na dva podskupa S1 i S2;
        // u S1 su tačke levo od duži AB, a u S2 su tačke desno od duži AB
        FindHull(S1, A, B);
        FindHull(S2, B, A);


FindHull(Sk, P, Q):
        ako skup Sk ne sadrži nijednu tačku, izaći iz funkcije;
        u skupu Sk pronaći najdalju tačku C od duži PQ;
        tačku C dodati u konveksni omotač;
        // neka je S1 skup tačaka sa leve strane duži PC;
        // neka je S2 skup tačaka sa leve strane duži CQ;
        FindHull(S1, P, C);
        FindHull(S2, C, Q);
```



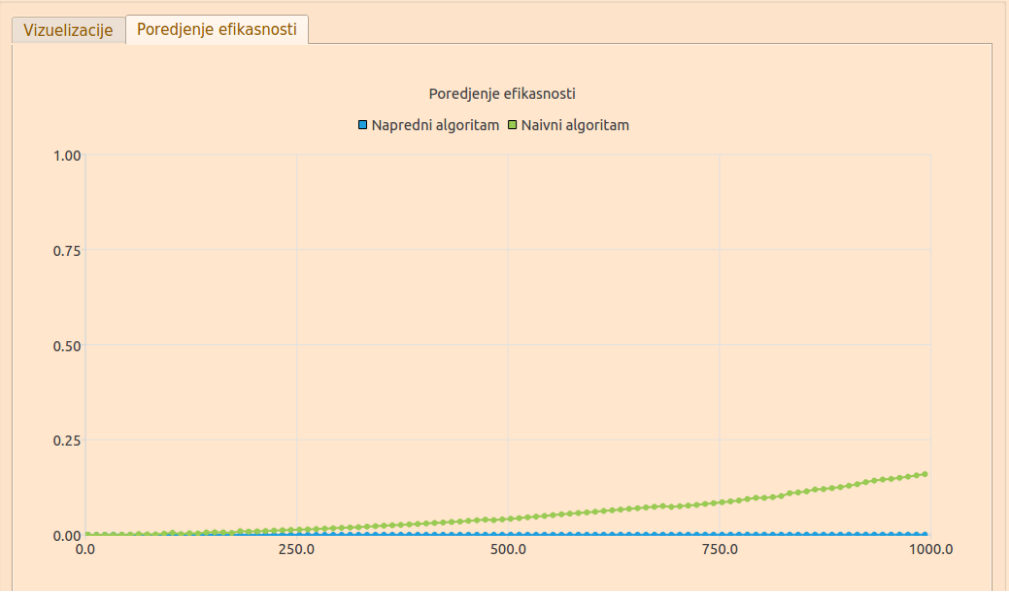
QuickHull algoritam

Na slici je prikazano određivanje tačke koja je najudaljenija od trenutne prave.

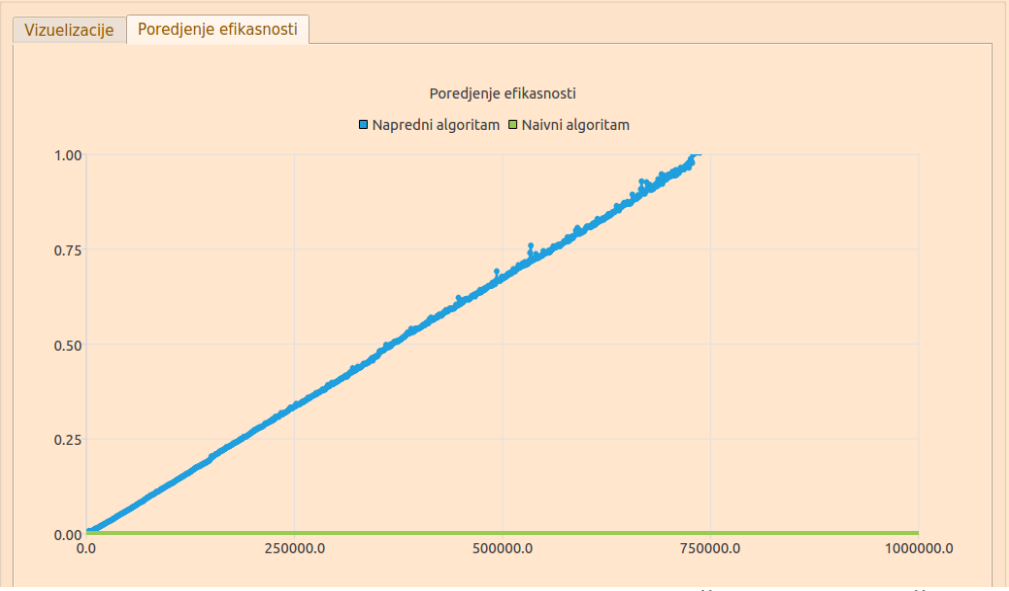
Poređenje efikasnosti naivnog i naprednih algoritama



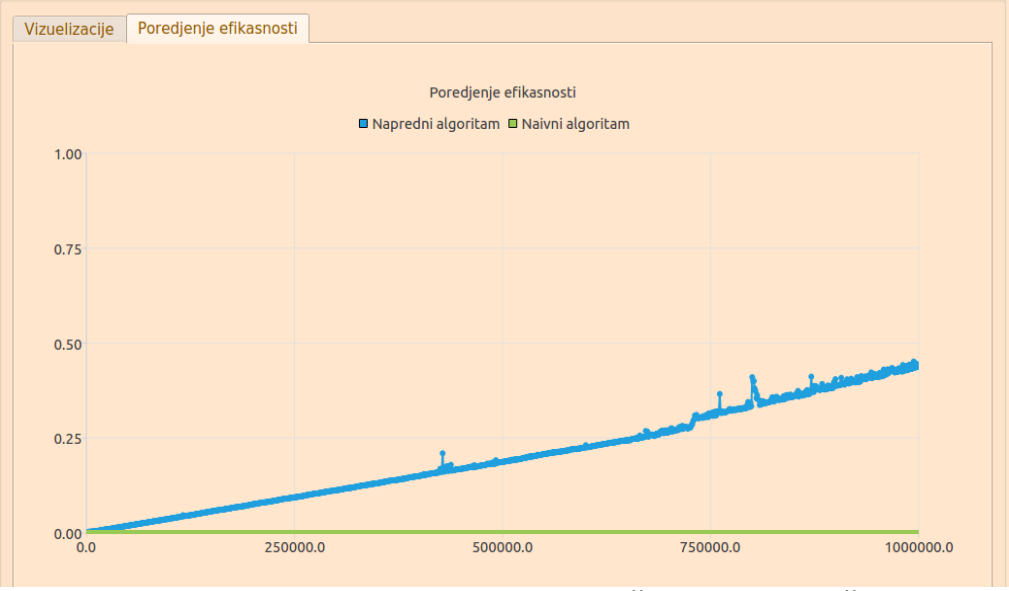
Inkrementalni i naivni algoritam - (maksimalna veličina skupa tačaka: 1.000; korak: 10)



QuickHull i naivni algoritam - (maksimalna veličina skupa tačaka: 1.000; korak: 10)



Inkrementalni algoritam - (maksimalna veličina skupa tačaka: 1.000.000; korak: 1000; napomena: brzina naivnog algoritma nije razmatrana)



QuickHull algoritam - (maksimalna veličina skupa tačaka: 1.000.000; korak: 1000; napomena: brzina naivnog algoritma nije razmatrana)

Testiranje ispravnosti algoritma

Većina test primera pokriva razne specijalne slučajeve gde je dato više kolinearnih tačaka.

Naziv testa	Opis testa	Ulaz	Očekivani izlaz
randomTest	Programski generisan ulaz	Niz dimenzije 30	Poklapanje rezultata naivnog i naprednih algoritama
validRandomTest1	Ručno generisan ulaz	100 200 150 160 200 140 213 84 130 300 240 325 300 200 400 200	Poklapanje rezultata naivnog i naprednog algoritma
validRandomTest2	Ručno generisan ulaz	100 200 150 160 200 140 213 84 300 84 400 84 130 300 240 325 300 200 400 200 150 200	Poklapanje rezultata naivnog i naprednih algoritama
validRectangleTest	Ručno generisan ulaz	50 50 50 100 50 150 100 50 100 100 100 150 150 50 150 100 150 150 200 50 200 100 200 150	Poklapanje rezultata naivnog i naprednih algoritama
duplicateTest	Nekoliko dupliranih	100 100 100 100	Poklapanje rezultata naivnog i

	tačaka	150 50 150 50 150 50 150 150 150 150 200 100 200 100 200 100 200 100	naprednih algoritama
lessThanThreePoints	Manje od tri tačke	25 53 243 592	Poklapanje rezultata naivnog i naprednih algoritama
validCollinearTest1	Kolinearne tačke	200 200 300 200 400 200 700 300	Poklapanje rezultata naivnog i naprednih algoritama
validCollinearTest2	Kolinearne tačke	200 200 300 200 400 200 700 50	Poklapanje rezultata naivnog i naprednih algoritama
validCollinearTest3	Kolinearne tačke	200 100 200 200 200 300 400 50	Poklapanje rezultata naivnog i naprednih algoritama
validCollinearTest4	Kolinearne tačke	200 100 200 200 200 300 400 400	Poklapanje rezultata naivnog i naprednih algoritama
validCollinearTest5	Kolinearne tačke	200 400 300 200 400 200 500 200	Poklapanje rezultata naivnog i naprednih algoritama
validCollinearTest6	Kolinearne tačke	100 50 200 200 300 200 400 200	Poklapanje rezultata naivnog i naprednih algoritama
validCollinearTest7	Kolinearne tačke	100 50 300 400 300 300 300 200	Poklapanje rezultata naivnog i naprednih algoritama

validCollinearTest8	Kolinearne tačke	100 600 300 400 300 300 300 200	Poklapanje rezultata naivnog i naprednih algoritama
validCollinearTest9	Kolinearne tačke	50 50 100 50 150 50 200 50 250 50	Poklapanje rezultata naivnog i naprednih algoritama
validCollinearTest10	Kolinearne tačke	50 250 100 200 150 150 200 100 250 50	Poklapanje rezultata naivnog i naprednih algoritama
validCollinearTest11	Kolinearne tačke	50 50 100 100 150 150 200 200 250 250	Poklapanje rezultata naivnog i naprednih algoritama
validCollinearTest12	Kolinearne tačke	50 50 50 100 50 150 50 200 50 250	Poklapanje rezultata naivnog i naprednih algoritama
validCollinearTest13	Kolinearne tačke	50 50 50 100 50 150 50 200 50 250 100 250 150 250 200 250 250 250 300 250	Poklapanje rezultata naivnog i naprednih algoritama
validCollinearTest14	Kolinearne tačke	250 50 250 100 250 150 250 200 250 250 50 250 100 250 150 250 200 250	Poklapanje rezultata naivnog i naprednih algoritama

validCollinearTest15	Kolinearne tačke	250 50 250 100 250 150 250 200 250 250 50 50 100 50 150 50 200 50	Poklapanje rezultata naivnog i naprednih algoritama
validCollinearTest16	Kolinearne tačke	50 50 50 100 50 150 50 200 50 250 100 50 150 50 200 50 250 50	Poklapanje rezultata naivnog i naprednih algoritama