Računanje s matricama u Android studiju

Galić, Ana

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

Sveučilišni studij

RAČUNANJE S MATRICAMA U ANDROID STUDIJU

Završni rad

Ana Galić

Osijek, 2023.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

Obrazac Z1P - Obrazac za ocjenu završnog rada na preddiplomskom sveučilišnom studiju

Osijek, 15.09.2023.

Odboru za završne i diplomske ispite

Prijedlog ocjene završnog rada na preddiplomskom sveučilišnom studiju

Ime i prezime Pristupnika:	Ana Galić
Studij, smjer:	Sveučilišni prijediplomski studij Elektrotehnika i informacijska
Mat. br. Pristupnika, godina upisa:	4821, 29.07.2020.
OIB Pristupnika:	81182099476
Mentor:	doc. dr. sc. Tomislav Rudec
Sumentor:	izv. prof. dr. sc. Alfonzo Baumgartner
Sumentor iz tvrtke:	
Naslov završnog rada:	Računanje s matricama u Android studiju
Znanstvena grana rada:	**Programsko inženjerstvo (zn. polje računarstvo)**
Zadatak završnog rad:	Studentica će u Android studiju napraviti grafičko sučelje za računanje operacija s matricama. Tema je zauzeta za Anu Galić. Sumentor s FERIT-a: Alfonzo Baumgartner.
Prijedlog ocjene završnog rada:	Dobar (3)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 1 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 2 bod/boda Jasnoća pismenog izražavanja: 2 bod/boda Razina samostalnosti: 2 razina
Datum prijedloga ocjene od strane mentora:	15.09.2023.
Datum potvrde ocjene od strane Odbora:	24.09.2023.

Potvrda mentora o predaji konačne verzije rada:	Mentor elektronički potpisao predaju konačne verzije.
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O ORIGINALNOSTI RADA

Osijek, 26.09.2023.

Ime i prezime studenta:	Ana Galić
Studij:	Sveučilišni prijediplomski studij Elektrotehnika i informacijska tehnologija
Mat. br. studenta, godina upisa:	4821, 29.07.2020.
Turnitin podudaranje [%]:	14

Ovom izjavom izjavljujem da je rad pod nazivom: **Računanje s matricama u Android studiju**

izrađen pod vodstvom mentora doc. dr. sc. Tomislav Rudec

i sumentora izv. prof. dr. sc. Alfonzo Baumgartner

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

Tema ovog završnog rada je izrada aplikacije u Android Studiju koja korisniku omogućava obavljanje matematičkih operacija s matricama putem mobilnog uređaja. Glavni cilj je pružiti jednostavan, ali učinkovit alat za izvođenje algebarskih operacija, podržavajući korisnike u procesu učenja tako što pruža lakšu i bržu provjeru rješenja zadataka povezanih s matricama. Izrada ove aplikacije zahtjeva znanje iz linearne algebre za uspješno razumijevanje i primjenu matematičkih operacija, što je ključno za razvoj funkcionalnosti aplikacije. Uz to je poznavanje iz programiranja nužno kako bi se aplikacija razvila u Android Studiju. Također važan aspekt koji treba uzeti u obzir je dizajn aplikacije i korisničkog sučelja kako bi korisnik mogao što jednostavnije koristiti aplikaciju i obavljati matematičke operacije bez poteškoća. Najprije će se proći kroz već postojeća slična rješenja koja će se usporediti te istaknuti prednosti i nedostatci. Zatim će se proći kroz osnove o matricama, matematičke operacije s matricama te će se detaljnije opisati posebni tipovi matrica. Nakon toga slijedi programsko rješenje aplikacije u kojem je opisan način izrade aplikacije, korišteni alati te prikazan izgled i funkcionalnost izrađene aplikacije.

1.1. Zadatak završnog rada

U ovom radu je cilj razviti grafičko sučelje koje će korisniku omogućiti obavljanje različitih računskih operacija s matricama. Operacije koje su korisniku dostupne su :

1. Zbrajanje matrica
2. Oduzimanje matrica
3. Množenje skalarom
4. Množenje dvije matrice
5. Pronalazak determinante

2. PREGLED POSTOJEĆIH RJEŠENJA

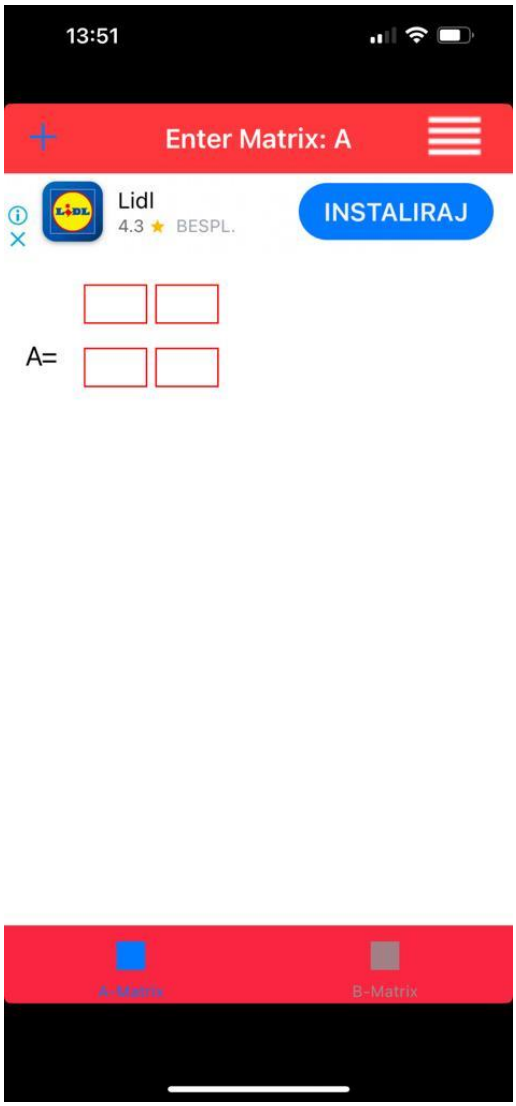
U slijedećem dijelu istražit ćemo i usporediti kakve sve aplikacije koje su napravljene kao kalkulator za matrice već postoje. Usporedit ćemo njihove karakteristike, ocijeniti jesu li funkcionalne, kakve mogućnosti pružaju, ima li kakvih naprednih opcija te istaknuti prednosti i nedostatke.

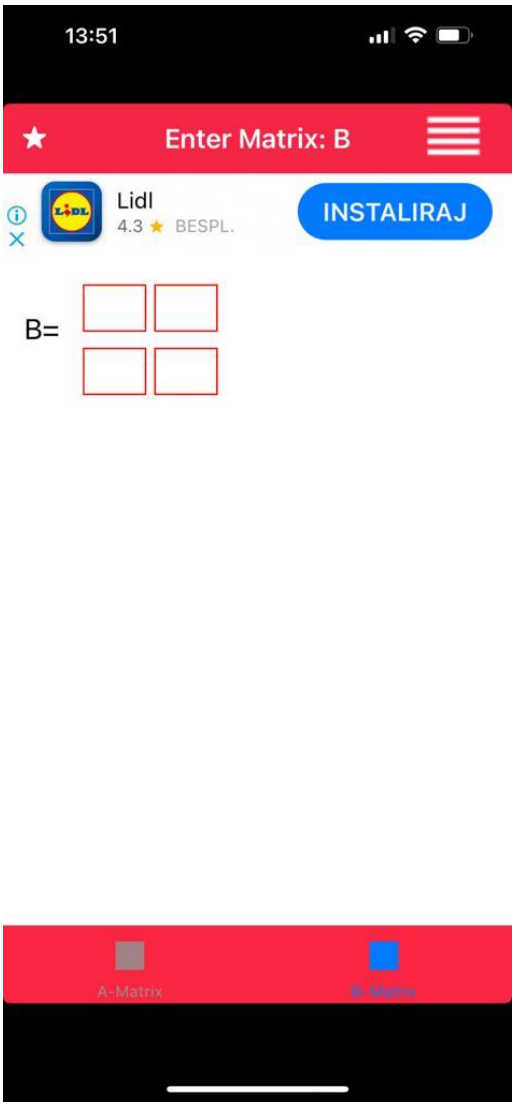
2.1. Matrix Calculator

Jedna od postojećih aplikacija koja je namijenjena za obavljanje operacija s matricama je Matrix Calculator. Aplikacija kao i sve druge ima svoje prednosti i mane. Pozitivna značajka je sposobnost obavljanja raznih matematičkih operacija s matricama. Operacije koje nudi :

- Zbrajanje
- Oduzimanje
- Dijeljenje
- Transponiranja

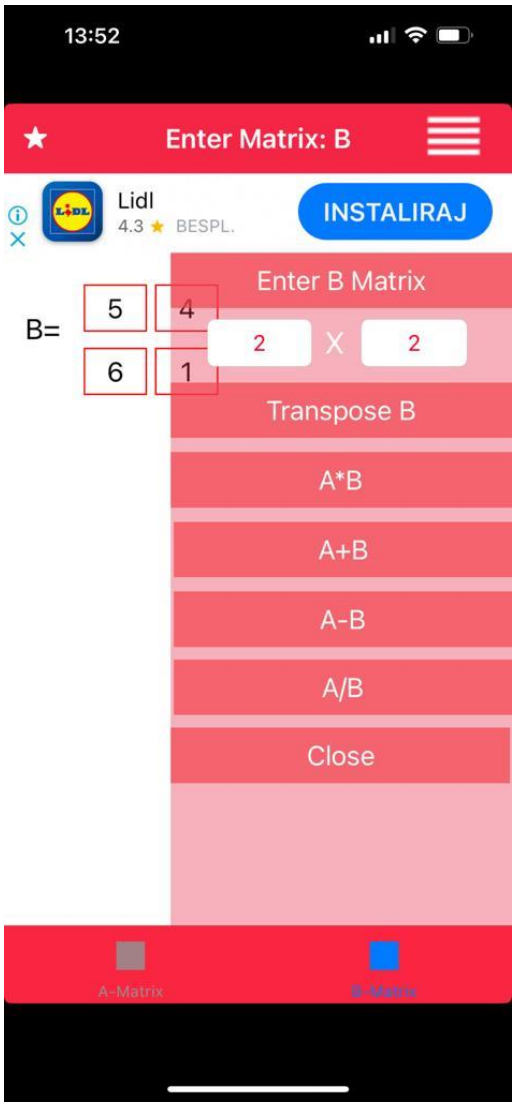
Još jedna velika prednost je to što korisniku nudi unos matrica dimenzija sve do 100x100. No, aplikacija je dostupna samo na iPhonima, što uvelike sužava mogućnost korištenja s obzirom da većina ljudi danas koristi Android mobilne uređaje. Dodatni veliki nedostatak je vizualni izgled. Pogoršava korisničko iskustvo jer su boje istaknute, spoj jarke crvene i tamno plave boje iritira oči te cijela aplikacija djeluje zastarjelo, nimalo moderno, što je i očekivano s obzirom da je stara 6 godina. Nedostatak jasnih uputa i loše organiziranog korisničkog sučelja može stvoriti poteškoće za korisnike. Prikaz aplikacije i njezinih funkcionalnosti nalazi se na slikama 2.1. – 2.4.



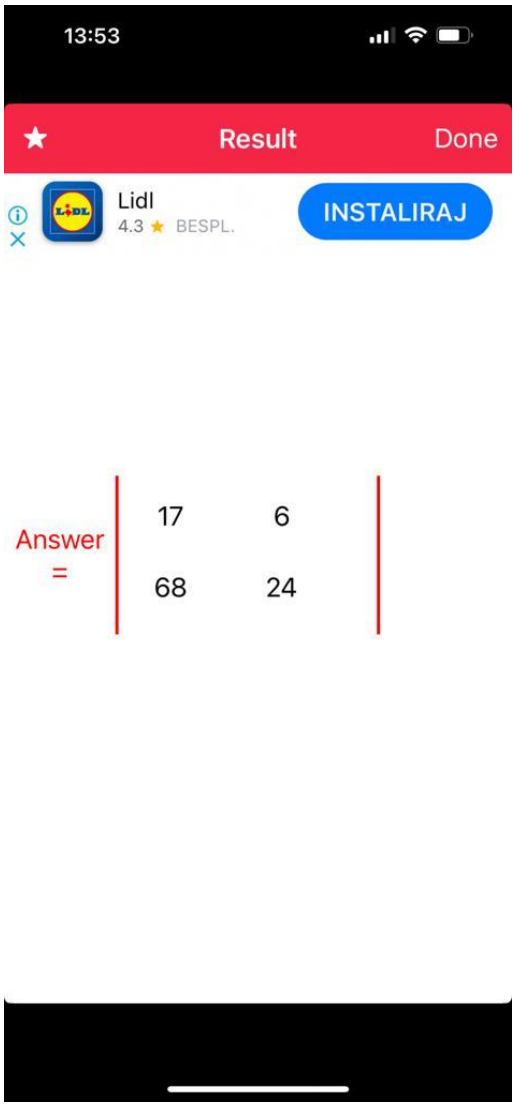


Sl. 2.1. Unos matrice A

Sl. 2.2. Unos matrice B



Sl. 2.3. Pregled mogućih operacija



Sl. 2.4. Konačno rješenje

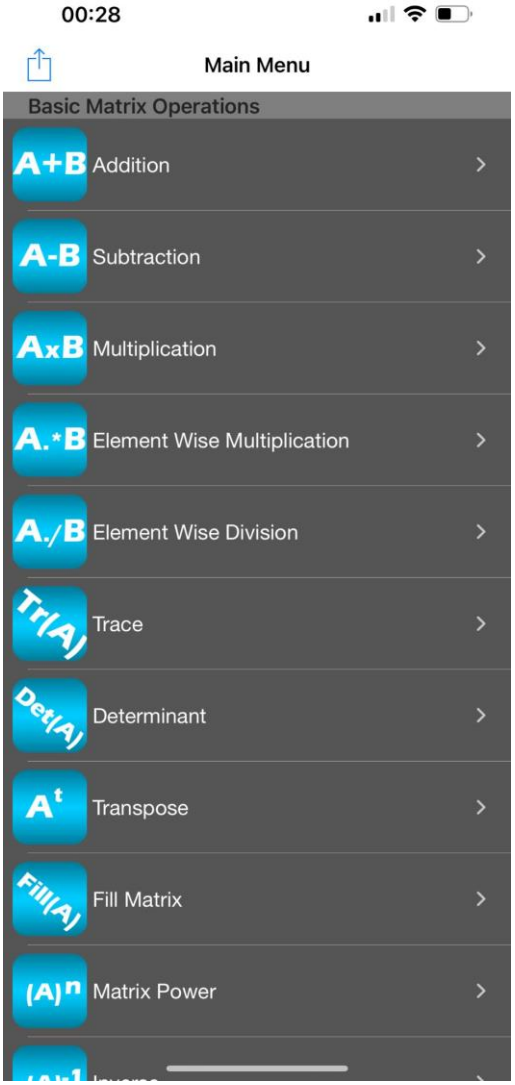
2.2. Matrix Calculator Plus

Još jedan primjer slične aplikacije je Matrix Calculator Plus. Aplikacija je vrlo dobro programski napravljena. Omogućuje još više računskih operacija nego prethodna. Korisniku nudi :

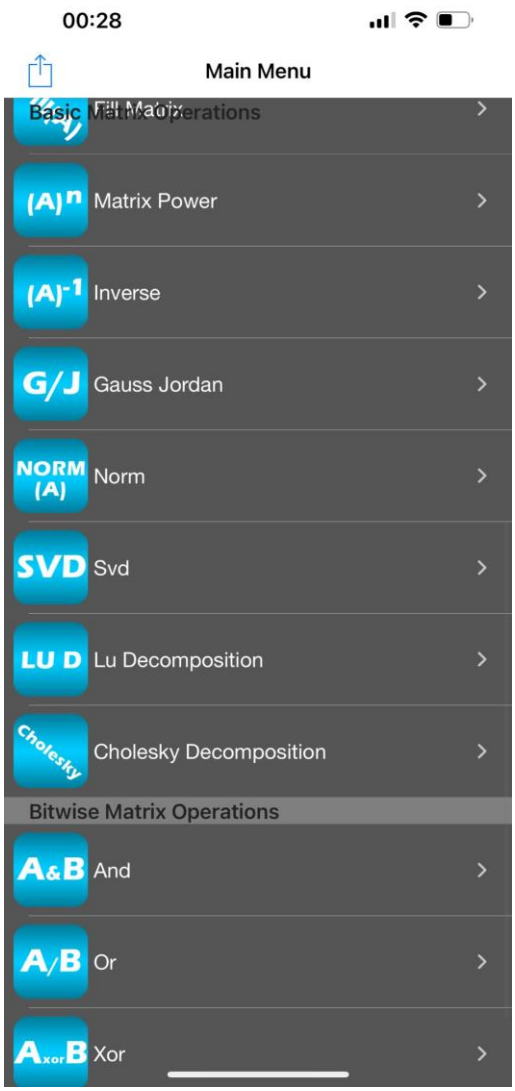
- Zbrajanje
- Oduzimanje
- Pronalazak determinante
- Pronalazak inverza
- Trag matrice
- Potenciranje matrice
- Gauss-Jordan eliminaciju

- I drugo

Najveća prednost ove aplikacije je programski dio, odnosno nudi mnoštvo računskih operacija, gotovo sve što je potrebno za rad s matricama. Napredna je što se vidi po kompleksnim operacijama koje nudi. Za razliku od primjera prošle aplikacije, moguće dimenzije matrica su nešto manje, nudi unos dimenzija do 20x20. Također, uspoređujući s prošlom, ima nešto kvalitetnije korisničko sučelje, no i dalje nema dojam moderne aplikacije. Očit je velik naglasak na programskom dijelu, za razliku od grafičkog dizajna. Prikaz aplikacije i njezinih funkcionalnosti nalazi se na slikama 2.5. – 2.9.

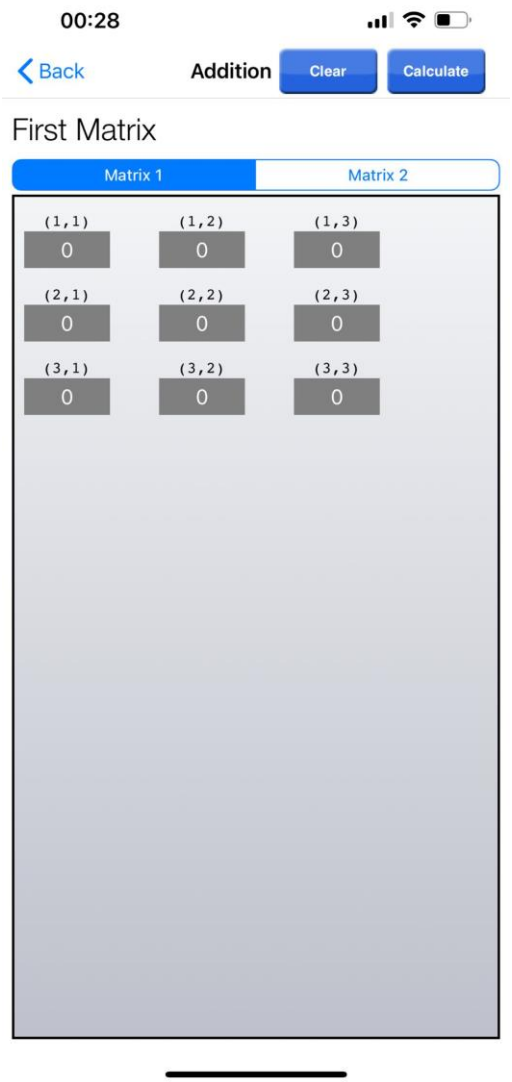


Sl. 2.5. Pregled mogućih operacija

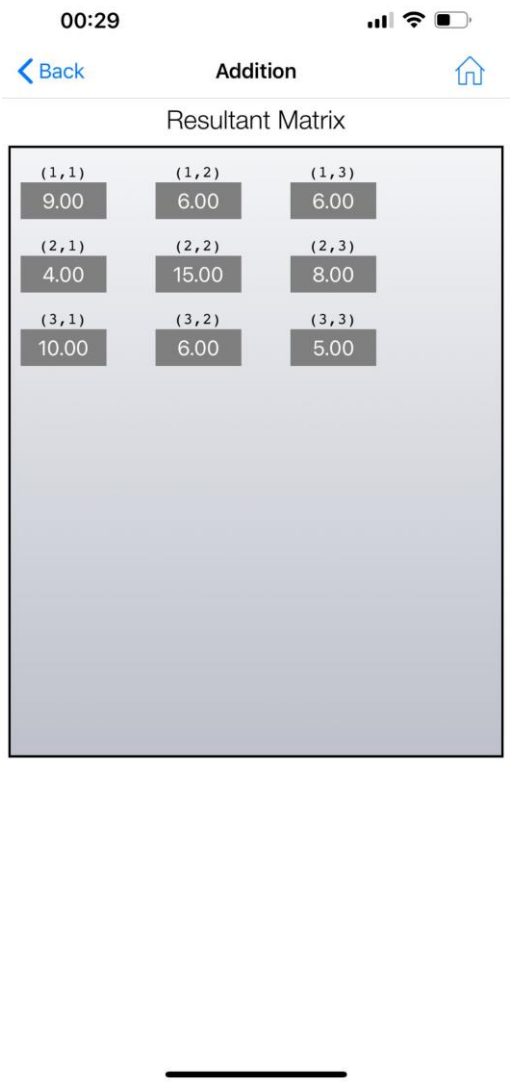


Sl. 2.6. Pregled mogućih operacija

Sl. 2.7. Unos dimenzija matrica



Sl. 2.8. Unos elemenata matrica



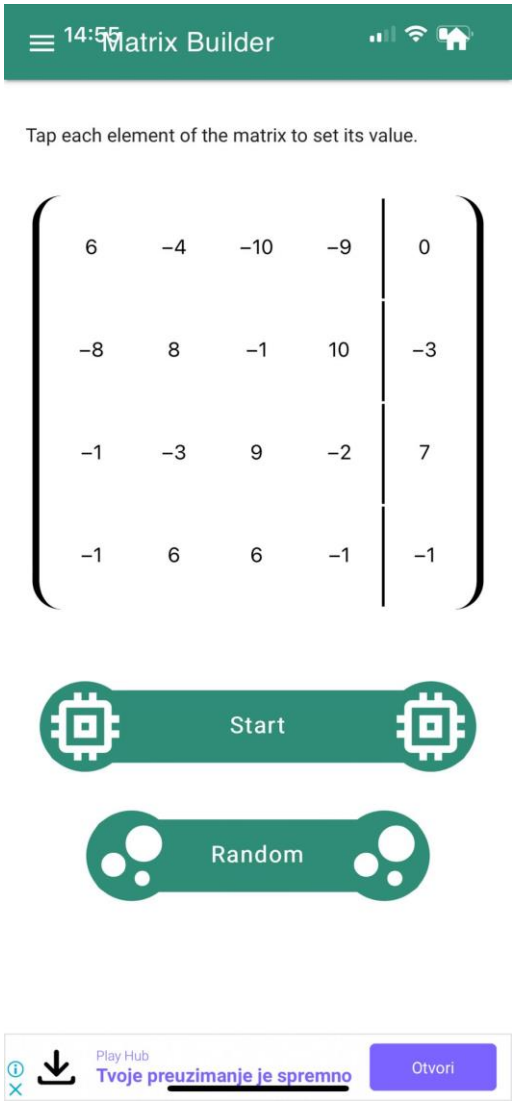
Sl. 2.9. Konačan rezultat

2.3. DIY Matrix Calculator

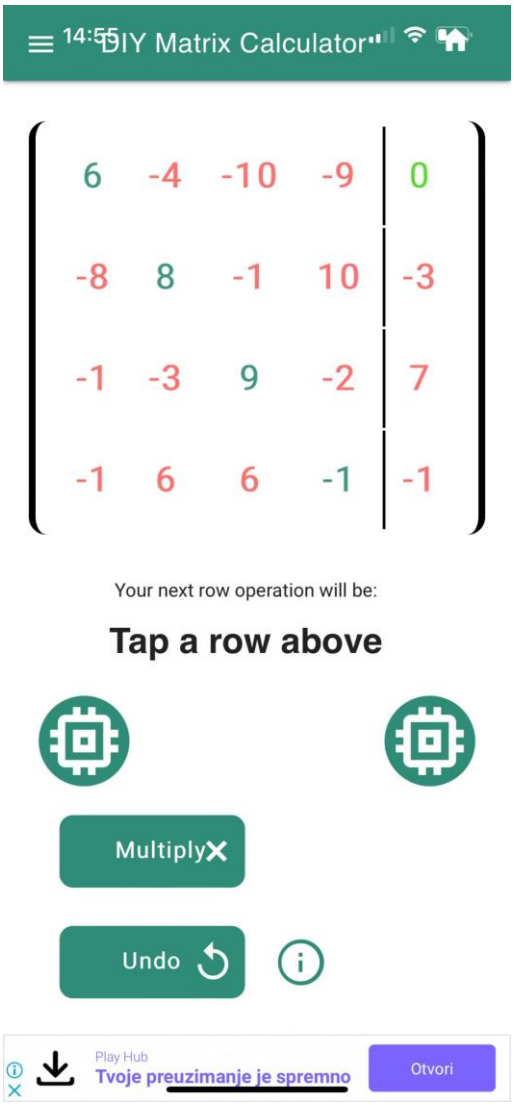
DIY Matrix Calculator je aplikacija sa namjenom kao i sve ostale do sad navedene, računanje matematičkih operacija s matricama. Napravljena je kao i sve tri do sad isključivo za iPhone, odnosno iOS sustave. Vizualno je bolja od prva dva primjera, ostavlja nešto moderniji dojam, ali tu prednosti staju. Uputstva za korištenje nisu sasvim jasna. Prilikom testiranja aplikacija nije radila, dopustila je samo unos dimenzija matrice te unos brojeva. Nakon toga su ponuđene samo dvije opcije računanja, a to je množenje i dijeljenje. Klikom na te dvije opcije ništa se ne događa

te vidimo da aplikacija zapravo radi uz poteškoće. Prikaz aplikacije i njezinih funkcionalnosti nalazi se na slikama 2.10. – 2.13.

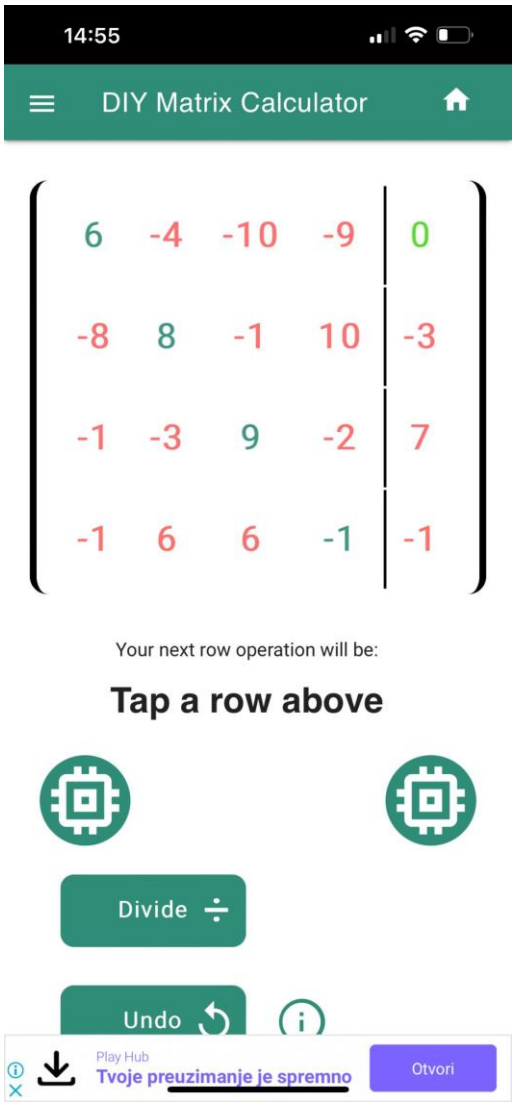
Sl. 2.10. Unos dimenzija matrice



Sl. 2.11. Unos elemenata matrice



Sl. 2.12. Odabir operacije



Sl. 2.13. Konačan rezultat

3. MATRICE

3.1. Osnove

Matrica je pravokutna tablica realnih brojeva sa *m* redaka i *n* stupaca, omeđenih uglatim zagradama. Svaki broj koji se nalazi u matrici predstavlja element matrice. Pripadajući elementi matrice ne odvajaju se zarezom, već samo praznim prostorom. Matrica sa m redaka i n stupaca naziva se m x n matrica ili matrica reda *(m,n).* Na slici 3.1. prikazan je općeniti oblik matrice, dok su na slici 3.2. prikazane matrice različitih dimenzija. [1].

$$A=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}, \quad m,n \in N$$

Sl. 3.1. Općenit oblik matrice

$$A=\underset{3\times 2}{\begin{bmatrix} 4 & 3 \\ 5 & 1 \\ 0 & -2 \end{bmatrix}} \qquad B=\underset{2\times 3}{\begin{bmatrix} -3 & 2 & 4 \\ 1 & -5 & 0 \end{bmatrix}} \qquad C=\underset{4\times 3}{\begin{bmatrix} 3 & -1 & 4 \\ 2 & 1 & 0 \\ -3 & 5 & 2 \\ 4 & -4 & 1 \end{bmatrix}}$$

Sl. 3.2. Primjeri izgleda matrice

Pozicija nekog elementa matrice označava se pripadnošću elementa određenom retku i stupcu, označava se notacijom s dvostrukim indeksima $a_{ij,}$ gdje *i* predstavlja broj retka, a *j* broj stupca. Matrica je kvadratna ukoliko vrijedi da je $m = n$. Elementi $a_{11,}\, a_{22,}\, \ldots,\, a_{nn}$ matrice čine glavnu dijagonalnu matrice. Ako matrici $A=(a_{ij})$ tipa $m\times n$, zamijenimo retke i stupce, dobit ćemo matricu $B=(b_{ij})$ tipa $n\times m$ pri čemu vrijedi $b_{ji}=b_{ij}$. Matricu *B* zovemo transponirana matrica matrice *A* i pišemo $B=A^T$.

3.2. Posebni tipovi realnih matrica

Simetrična matrica je kvadratna matrica koja je simetrična s obzirom na glavnu dijagonalu, tj. vrijedi : $a_{ji} = a_{ij}, \forall i,j \in \{1,\ldots,n\}$. Za simetričnu matricu A vrijedi: $A^T = A$. Primjer takve matrice nalazi se na slici 3.3.

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 6 & 4 \\ 3 & 4 & 0 \end{bmatrix}$$

Sl. 3.3. Primjer simetrične matrice

Antisimetrična matrica je kvadratna matrica za koju vrijedi da je $a_{ij} = -a_{ji}$. Odnosno vrijedi uvjet : $a_{ji} = -a_{ij}, \ \forall i,j \in \{1,\ldots,n\}$. Za simetričnu matricu A vrijedi: $A^T = -A$. Primjer takve matrice prikazan je na slici 3.4.

$$B = \begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 4 \\ -2 & -4 & 0 \end{bmatrix}$$

Sl. 3.4. Primjer antisimetrične matrice

Dijagonalna matrica je kvadratna matrica u kojoj su svi elementi izvan glavne dijagonale jednaki nuli. Odnosno vrijedi uvjet : $a_{ij} = 0$, za sve $i \neq j$. Primjer takve matrice prikazan je na slici 3.5.

$$D = \begin{bmatrix} 2 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 3 \end{bmatrix}$$

Sl. 3.5. Primjer dijagonalne matrice

Jedinična matrica je kvadratna matrica koja ima n redaka i n stupaca i kod koje se jedinice nalaze na glavnoj dijagonali. Vrijedi uvjet : $a_{ij} = 1$ za $i = j$, $a_{ij} = 0$ za $i \neq j$. Primjer takve matrice prikazan je na slici 3.6.

$$I = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Sl. 3.6. Primjer jedinične matrice

Nulmatrica je matrica kod koje su svi elementi matrice jednaki nuli. Primjer takve matrice prikazan je na slici 3.7.

$$O = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Sl. 3.7. Primjer nulmatrice

Trokutasta matrica je kvadratna matrica koja ima sve elemente iznad glavne dijagonale jednake nuli (donja trokutasta) ili sve elemente ispod glavne dijagonale jednake nuli (gornja trokutasta). Primjer takve matrice prikazan je na slici 3.8.

$$T = \begin{bmatrix} 3 & 4 & -2 & 1 \\ 0 & 1 & -9 & 7 \\ 0 & 0 & 5 & 6 \\ 0 & 0 & 0 & -1 \end{bmatrix}$$

Sl. 3.8. Primjer trokutaste matrice

Jednoredna matrica ili vektor redak je naziv za matricu koja ima samo jedan redak. Jednostupčana matrica ili vektor stupac je matrica koja ima samo jedan stupac. Primjer jednoredne matrice prikazan je na slici 3.9., a jednostupčane na slici 3.10.

$$A = \begin{bmatrix} -7 & 3 & -89 \end{bmatrix}$$

Sl. 3.9. Primjer jednoredne matrice

$$A = \begin{bmatrix} -6 \\ 6 \\ 9 \end{bmatrix}$$

Sl. 3.10. Primjer jednostupčane matrice

3.3. Matematičke operacije s matricama

3.3.1. Zbrajanje i oduzimanje matrica

Dvije matrice se mogu zbrojiti odnosno oduzeti ako vrijedi da su to matrice istog reda. Obavlja se tako da se zbroje odnosno oduzmu elementi koji se nalaze na istim pozicijama u obje matrice. Primjer postupka zbrajanja dviju matrica nalazi se na slici 3.11.

Svojstva zbrajanja matrica :

1. Asocijativnost zbrajanja : $(A + B) + C = A + (B + C)$, za sve matrice $A, B, C \in M_{mn}(R)$.
2. Postojanje neutralnog elementa : $A + N = N + A = A$, za sve matrice $A \in M_{mn}(R)$, $N \in M_{mn}(R)$
3. Postojanje suprotnog elementa : $A + B = B + A = N$, za svaku matricu $A \in M_{mn}(R)$ postoji $B \in M_{mn}(R)$
4. Komutativnost zbrajanja : $A + B = B + A$, za sve matrice $A, B \in M_{mn}(R)$

$$\begin{bmatrix} 1 & 3 & 2 \\ 1 & 0 & 0 \\ 1 & 2 & 2 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 5 \\ 7 & 5 & 0 \\ 2 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1+0 & 3+0 & 2+5 \\ 1+7 & 0+5 & 0+0 \\ 1+2 & 2+1 & 2+1 \end{bmatrix} = \begin{bmatrix} 1 & 3 & 7 \\ 8 & 5 & 0 \\ 3 & 3 & 3 \end{bmatrix}$$

Sl. 3.11. Primjer zbrajanja matrica

3.3.2. Množenje matrica sa skalarom

Množenje matrice sa skalarom obavlja se tako da se svaki element matrice pomnoži sa zadanim brojem, odnosno skalarom k. Skalar može time povećati ili smanjiti matricu u određenom razmjeru te može biti i negativan broj. Matrica bilo kojeg reda može se množiti sa skalarom. Primjer takvog postupka nalazi se na slici 3.12.

Svojstva koja vrijede kod množenja sa skalarom su :

1. Distributivnost u odnosu na zbrajanje matrica : $\lambda(A + B) = \lambda A + \lambda B$
2. Distributivnost u odnosu na zbrajanje skalara : $(\lambda + \mu)A = \lambda A + \mu A$
3. Kvaziasocijativnost : $(\lambda \cdot \mu)A = \lambda(\mu A)$

$$2 \cdot \begin{bmatrix} 10 & 6 \\ 4 & 3 \end{bmatrix} = \begin{bmatrix} 2 \cdot 10 & 2 \cdot 6 \\ 2 \cdot 4 & 2 \cdot 3 \end{bmatrix}$$

Sl. 3.12. Primjer množenja matrice skalarom

3.3.3. Množenje matrica

Množenje dviju matrica *A* i *B* je moguće samo ukoliko matrica *A* ima toliko stupaca koliko matrica *B* ima redaka, odnosno da su matrice ulančane. Primjer takvog postupka nalazi se na slici 3.13. Neka je matrica *A* tipa *m X k*, a matrica *B* tipa *k X n*, tada je matrica $C = A \cdot B$ tipa *m X n* i vrijedi: $c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \ldots + a_{ik}b_{kj}$

$$\begin{bmatrix} 1 & 0 & 2 \\ -1 & 3 & 1 \end{bmatrix} \cdot \begin{bmatrix} 3 & 1 \\ 2 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} (1 \cdot 3 + 0 \cdot 2 + 2 \cdot 1) & (1 \cdot 1 + 0 \cdot 1 + 2 \cdot 0) \\ ((-1) \cdot 3 + 3 \cdot 2 + 1 \cdot 1) & ((-1) \cdot 1 + 3 \cdot 1 + 1 \cdot 0) \end{bmatrix}$$

Sl. 3.13. Primjer množenja matrica

3.3.4. Transponiranje matrica

Transponiranje matrice postiže se zamjenom njezinih redaka i stupaca. Ako je A matrica tipa (n, m) s elementima a_{ij}, njezina transponirana matrica A^T ima elemente $(A^T)_{nm} = a_{mn}$, što se kod kvadratne matrice može interpretirati i kao zrcaljenje s obzirom na dijagonalu. Primjer takvog postupka nalazi se na slici 3.14.

$$A = \begin{bmatrix} 1 & 2 \\ -3 & 1 \\ 0 & -2 \end{bmatrix} \qquad A^T = \begin{bmatrix} 1 & -3 & 0 \\ 2 & 1 & -2 \end{bmatrix}$$

Sl. 3.14. Primjer transponiranja matrice

3.3.5. Determinanta matrice

Determinanta matrice predstavlja broj koji je pridružen kvadratnoj matrici. Označava se simbolom $|A|$ ili $det(A)$, gdje je A kvadratna matrica. Izračunava se na više načina, pomoću Laplacea, Gaussove eliminacije, LU dekompozicije, Sarrusovog pravila i drugih metoda. [4]. Svojstva determinante :

- Ako vrijedi da je $det(A) = 0$, matrica A je singularna, tj. nema inverznu matricu
- Ako vrijedi da je $det(A) \neq 0$, matrica A ima inverznu matricu
- Determinanta matrice A jednaka je determinanti transponirane matrice $det(A) = det(A^T)$
- Determinanta umnoška dviju matrica jednaka je umnošku njihovih determinanti $det(AB) = det(A) * det(B)$

3.3.6. Inverz matrice

Inverzna matrica je matrica čija su oba umnoška nje i zadane matrice jednaka jediničnoj matrici. Inverzna matrica mora biti kvadratna i nenul matrica. Inverzna matrica nije definirana za singularne matrice, tj. matrice čija je determinanta jednaka nuli. [5]. Primjer računanja inverza matrice nalazi se na slici 3.15. Inverz matrice određuje se sljedećim postupkom :

1. Računanje determinante matrice
2. Određivanje transponirane matrice
3. Svaki element transponirane matrice zamjenjuje se sa odgovarajućim kofaktorom i rezultat je adjungirana matrica
4. Određuje se inverz matrice

$$A = \begin{bmatrix} 2 & 4 \\ 8 & 3 \end{bmatrix}$$

$$\det(A) = 2 \cdot 3 - 4 \cdot 8 = -26$$

$$A_{11} = (-1)^{1+1} \cdot 3 = 3$$

$$A_{12} = (-1)^{1+2} \cdot 8 = -8$$

$$A_{21} = (-1)^{2+1} \cdot 4 = -4$$

$$A_{22} = (-1)^{2+2} \cdot 2 = 2$$

$$A^{-1} = \frac{1}{-26} \cdot \begin{bmatrix} 3 & -8 \\ -4 & 2 \end{bmatrix}^T$$

$$A^{-1} = \begin{bmatrix} \frac{-3}{26} & \frac{2}{13} \\ \frac{4}{13} & \frac{-1}{13} \end{bmatrix}$$

Sl. 3.15. Primjer računanja inverza matrice

4. PROGRAMSKO RJEŠENJE APLIKACIJE

4.1. Korištene tehnologije

Tehnologija korištena za izradu aplikacije kalkulatora s matricama zove se Android Studio. Android Studio je integrirano razvojno okruženje (engl. *Integrated development environment* - IDE) za razvoj Android aplikacija, detaljnije objašnjeno u [6]. Predstavlja pomoćni alat za izradu, testiranje i pokretanje aplikacija. Dostupan je za preuzimanje na Windows, macOS i Linux operativnim sustavima. Podržava programske jezike kao što su: Java, C++ i Kotlin, a upravo je on i korišten za izradu ove aplikacije. U okviru Android Studija koristi se XML (engl. *Extensible Markup Language*) koji predstavlja jezik za označavanje podataka, sličan je HTML-u (engl. *HyperText Markup Language*), detaljnije objašnjeno na [7]. Osim Android Studija korištena je i Figma, odnosno alat za dizajn koji korisnicima pomaže vizualno izraziti svoje ideje. Omogućuje stvaranje, dijeljenje i testiranje dizajna za web stranicu, mobilnu aplikaciju i druge digitalne proizvode.

4.2. Prikaz programskog rješenja

4.2.1. Početna animacija

Prilikom otvaranja aplikacije prikazuje se animacija napravljena u već spomenutoj Figmi. Animacija je napravljena pomoću 9 *frameova*, odnosno elemenata koji služe kao prikaz okvira uređaja, te pametne interakcije odnosno animacije između svakog od njih. Za izvoz takve animacije potreban je dodatan alat pod nazivom Aninix UI Animation koji omogućuje povezivanje svih pokreta animacije, uređivanje te izvoz u Lottie te razne druge medijske formate. Pomoću Aninix UI Animation alata programeri mogu lako sve izrađene animacije ugraditi u proizvod, odnosno projekt. Na slici 4.1. prikazan je kod pomoću kojeg se prikazuje animacija.

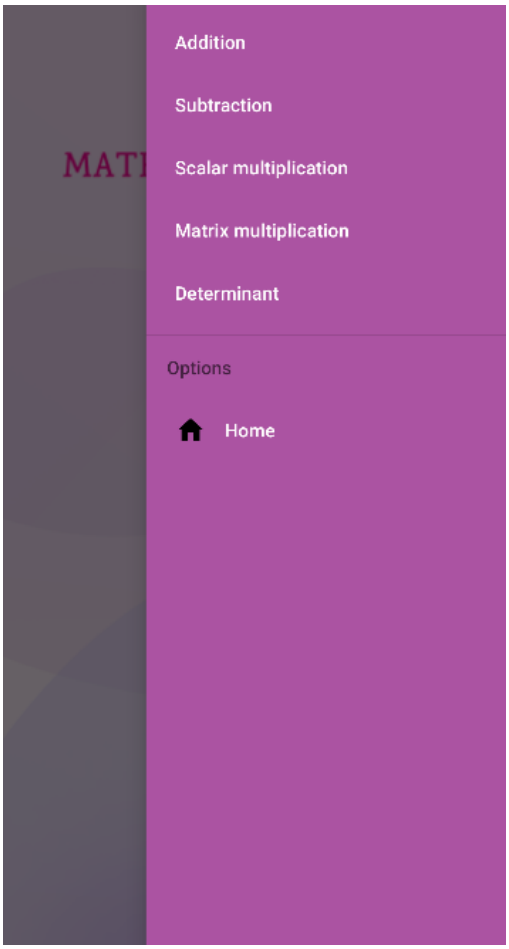
```
val videoView = findViewById<VideoView>(R.id.videoView)
val videoPath = "android.resource://${packageName}/${R.raw.animation}"
val videoUri = Uri.parse(videoPath)

videoView.setVideoURI(videoUri)
videoView.setOnPreparedListener { mediaPlayer ->
    mediaPlayer.isLooping = false
    mediaPlayer.start()
}

videoView.setOnCompletionListener {  it: MediaPlayer!
    val fragment = MainFragment()
    val transaction = supportFragmentManager.beginTransaction()

    transaction.replace(com.google.android.material.R.id.container, fragment)
    transaction.addToBackStack( name: null)
    transaction.commit()
}
```

Sl. 4.1. Prikaz animacije na početnom zaslonu

4.2.2. Početni zaslon

Početni zaslon prikazan na slici 4.2. prikazuje 4 gumba kao mogućnost unosa 4 različite matrice, odnosno matrice A, B, C i D. Sa desne strane zaslona nalazi se bočna traka sa mogućnosti odabira 5 različitih matematičkih operacija : zbrajanje matrica, oduzimanje matrica, množenje skalarom, množenje dvije matrice te računanje determinante matrice kao što je prikazano na slici 4.3. Uz navedenih 5 matematičkih operacija nalazi se dodatna opcija Home, odnosno mogućnost vraćanja na početni zaslon. Bočna traka također se nalazi na svakom zaslonu, odnosno fragmentu aplikacije.

Sl. 4.2. Prikaz početnog zaslona



Sl. 4.3. Prikaz bočne trake

```
matrix_a_Btn?.setOnClickListener{ it: View!
    matrixSingleton.currentMatrix = "A"
    if (isCreated(matrixSingleton.matrixA)){

        val fragmentTransaction: FragmentTransaction? =
            activity?.supportFragmentManager?.beginTransaction()
        fragmentTransaction?.replace(R.id.fragmentContainerView, editMatrixFragment)
        fragmentTransaction?.commit()
    }
    else{
        val fragmentTransaction: FragmentTransaction? =
            activity?.supportFragmentManager?.beginTransaction()
        fragmentTransaction?.replace(R.id.fragmentContainerView, fragmentDimension)
        fragmentTransaction?.commit()
    }

}
```

Sl. 4.4. Zamjena fragmenta klikom na gumb matrice

Sl. 4.5. Prikaz bočne trake i promjena odgovarajućeg fragmenta

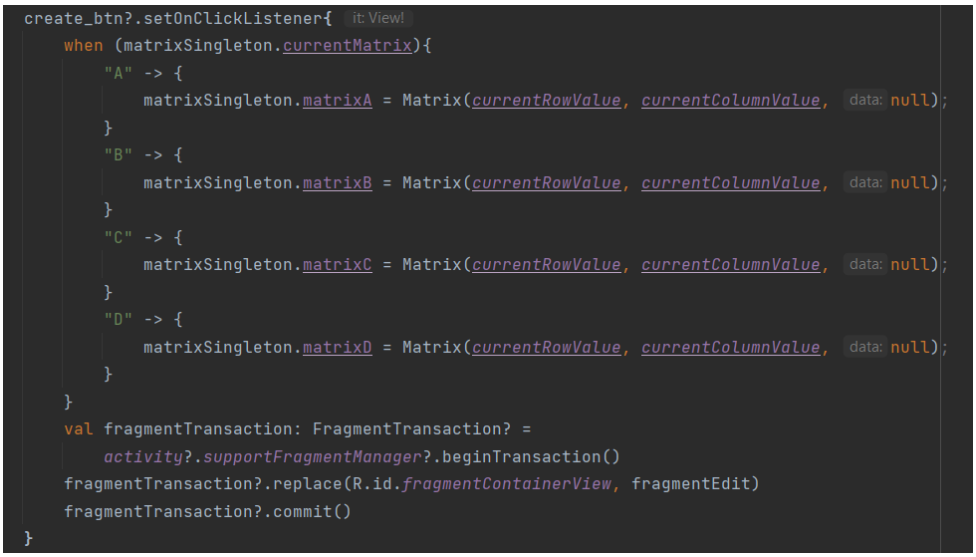
4.2.3. Unos dimenzija

Zaslon za unos dimenzija nalazi se na slici 4.6. i sastoji se od 4 gumba : *add row, add column, create* i strelice za vraćanje nazad. *Add row i add column* gumb dodaju redove odnosno stupce, ovisno koliko puta na gumb kliknemo. Broj redaka ili stupaca može biti od 1 do 4. Kada korisnik četiri puta pritisne gumb, broj redaka ili stupaca se automatski resetira na 1, i svakim dodatnim pritiskom na gumb dodaje se još jedan redak ili stupac. Ta funkcionalnost je prikazana na slici 4.7. *Create* gumb sprema prijašnje unesene dimenzije i odvodi na novi fragment, odnosno na zaslon za unos elemenata te iste matrice. Kod navedene funkcionalnosti prikazan je na slici 4.8. Klikom na gumb sa strelicom za vraćanje nazad odvodi na prijašnji fragment, odnosno početni zaslon.

Sl. 4.6. Zaslon za unos dimenzija matrice

```
btnAddRow.setOnClickListener {  it: View!
    currentRowValue += 1
    if (currentRowValue > 4) {
        currentRowValue = 1
    }
    row_input.text = currentRowValue.toString()
}

btnAddColumn.setOnClickListener {  it: View!
    currentColumnValue += 1

    if (currentColumnValue > 4) {
        currentColumnValue = 1
    }

    column_input.text = currentColumnValue.toString()
}
```

Sl. 4.7. Dodavanje redaka i stupaca matrice



Sl. 4.8. Spremanje vrijednosti redaka i stupaca te mijenjanje fragmenta klikom na gumb Create

4.2.4. Unos elemenata

Zaslon za unos elemenata sa slike 4.9. sastoji se od dva gumba, *delete* i *confirm*, jednog *grid layouta* koji se dodatno sastoji od onoliko *EditTextova* koliko ima elemenata matrice te 2 *ImageViewa*. *ImageView* je element XML-a koji označava sliku. U ovom fragmentu te dvije slike predstavljaju uglate zagrade matrice. *EditText* je element XML-a koji omogućuje unos teksta. *Grid layout* predstavlja razmještaj elemenata u obliku mreže koja se sastoji od redaka i stupaca. Elementi koji se unose će se spremiti u odgovarajuću, ranije odabranu matricu, što je prikazano na slici 4.10. Ukoliko je već prijašnje unesena matrica te se stisnulo na gumb *confirm*, matrica se spremila, način na koji se to napravilo nalazi se na slici 4.11., te klikom na gumb te iste matrice na početnom zaslonu vraća nas na upravo ovaj zaslon za unos elemenata te su prikazani već prijašnje spremljeni elementi matrice. *Delete* gumb briše već unesenu matricu te nas vraća na zaslon za unos dimenzija, kako bi mogli ponovno unijeti nove dimenzije i nove elemente. Kod te funkcionalnost je prikazan na slici 4.12.

Sl.4.9. Prikaz zaslona za unos elemenata matrice

```
when (matrixSingleton.currentMatrix){
    "A" -> {
        numRows = matrixSingleton.matrixA?.rows!!
        numCols = matrixSingleton.matrixA?.cols!!
        if (matrixSingleton.matrixA != null) {
            data = matrixSingleton.matrixA!!.data
        }
    }
    "B" -> {
        numRows = matrixSingleton.matrixB?.rows!!
        numCols = matrixSingleton.matrixB?.cols!!
        if (matrixSingleton.matrixB != null) {
            data = matrixSingleton.matrixB!!.data
        }
    }
    "C" -> {
        numRows = matrixSingleton.matrixC?.rows!!
        numCols = matrixSingleton.matrixC?.cols!!
        if (matrixSingleton.matrixC != null) {
            data = matrixSingleton.matrixC!!.data
        }
    }
    "D" -> {
        numRows = matrixSingleton.matrixD?.rows!!
        numCols = matrixSingleton.matrixD?.cols!!
        if (matrixSingleton.matrixD != null) {
            data = matrixSingleton.matrixD!!.data
        }
    }
}
```

Sl. 4.10. Odabir odgovarajuće matrice za popunjene elemenate

```
confirm_btn.setOnClickListener {  it: View!
    val numRows = matrixData.size
    val numCols = if (numRows > 0) matrixData[0].size else 0

    val dataList = mutableListOf<List<Int>>()

    for (row in 0 ≤ until  < numRows) {
        val rowList = mutableListOf<Int>()
        for (col in 0 ≤ until  < numCols) {
            val editText = matrixData[row][col]
            val value = editText.text.toString().toIntOrNull() ?: 0
            rowList.add(value)
        }
        dataList.add(rowList)
    }
    val matrix = Matrix(numRows, numCols, dataList)

    when (matrixSingleton.currentMatrix){
        "A" -> matrixSingleton.matrixA = matrix
        "B" -> matrixSingleton.matrixB = matrix
        "C" -> matrixSingleton.matrixC = matrix
        "D" -> matrixSingleton.matrixD = matrix
    }

    val fragmentTransaction: FragmentTransaction? =
        activity?.supportFragmentManager?.beginTransaction()
    fragmentTransaction?.replace(R.id.fragmentContainerView, mainFragment)
    fragmentTransaction?.commit()
}
```

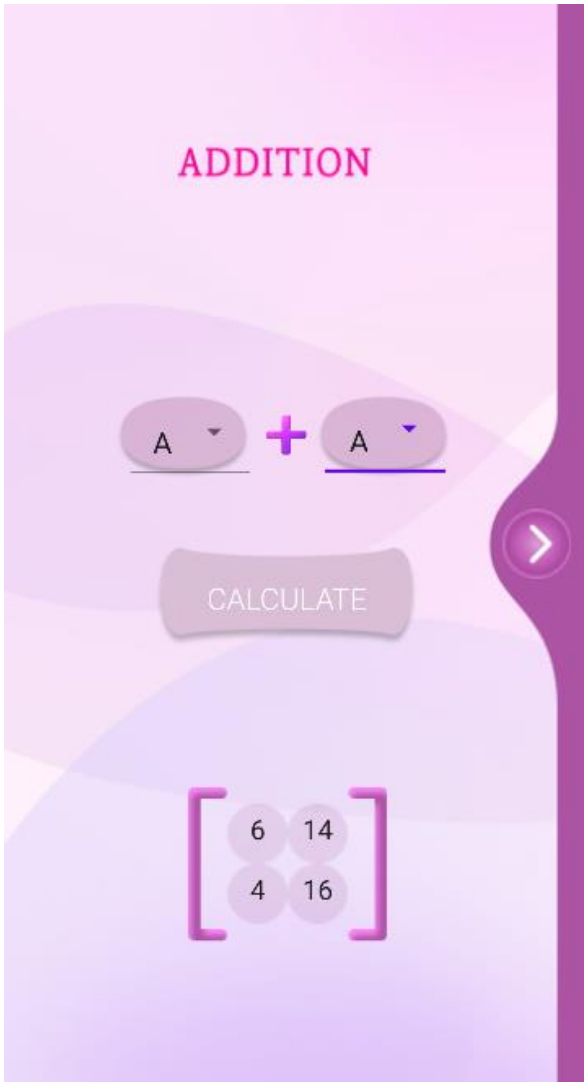
Sl. 4.11. Spremanje elemenata odgovarajuće matrice

```
delete_btn.setOnClickListener {  it: View!
    when (matrixSingleton.currentMatrix){
        "A" -> matrixSingleton.matrixA = null
        "B" -> matrixSingleton.matrixB = null
        "C" -> matrixSingleton.matrixC = null
        "D" -> matrixSingleton.matrixD = null
    }

    val fragmentTransaction: FragmentTransaction? =
        activity?.supportFragmentManager?.beginTransaction()
    fragmentTransaction?.replace(R.id.fragmentContainerView, mainFragment)
    fragmentTransaction?.commit()
}
```

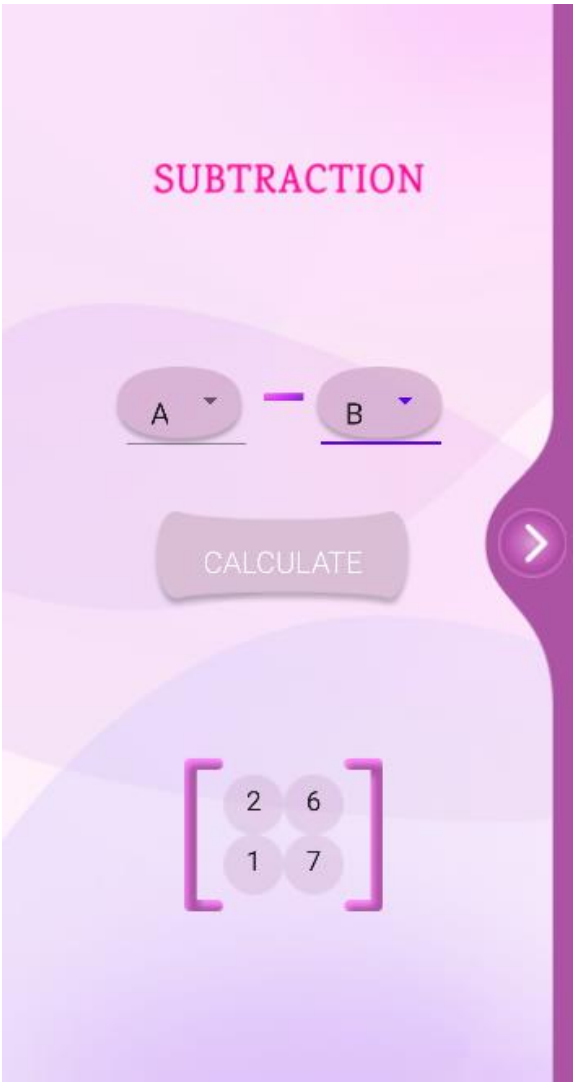
Sl. 4.12. Brisanje svih elemenata matrice

4.2.5. Zbrajanje i oduzimanje matrica

Fragmenti za zbrajanje, odnosno oduzimanje matrica prikazani su na slici 4.13. i 4.14. Sastoje se od već spomenutog *grid layouta*, gumba *calculate*, 3 *ImageViewa* te 2 padajuća izbornika. Gumb *calculate* prikazuje unutar *grid layouta* novu matricu koja se dobije kao rezultat zbrajanja, odnosno oduzimanja dvije matrice. Padajući izbornici pružaju 4 mogućnosti odabira, a to su matrice A, B, C i D. Operacije zbrajanja i oduzimanja matrica prikazane su na slikama 4.15. i 4.16.



Sl. 4.13. Zbrajanje dvije matrice



Sl.4.14. Oduzimanje dvije matrice

```
private fun addMatrices(matrix1: List<List<Int>>, matrix2: List<List<Int>>): List<List<Int>> {
    val numRows = matrix1.size
    val numCols = matrix1[0].size

    val result = MutableList(numRows) { MutableList(numCols) { 0 } }

    for (i in 0 ≤ until < numRows) {
        for (j in 0 ≤ until < numCols) {
            result[i][j] = matrix1[i][j] + matrix2[i][j]
        }
    }
    return result
}
```

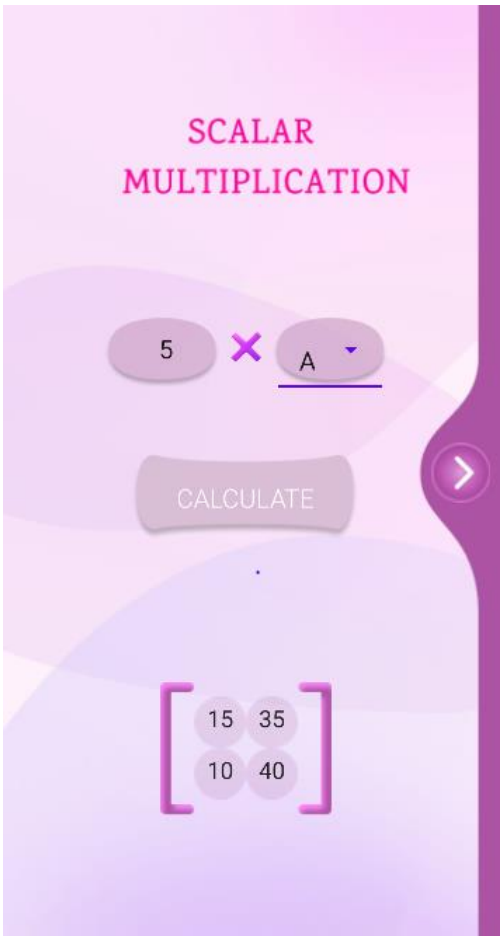
Sl. 4.15. Računanje zbrajanja matrica

```
private fun subtractMatrices(matrix1: List<List<Int>>, matrix2: List<List<Int>>): List<List<Int>> {
    val numRows = matrix1.size
    val numCols = matrix1[0].size

    val result = MutableList(numRows) { MutableList(numCols) { 0 } }

    for (i in 0 ≤ until < numRows) {
        for (j in 0 ≤ until < numCols) {
            result[i][j] = matrix1[i][j] - matrix2[i][j]
        }
    }
    return result
}
```

Sl. 4.16. Računanje oduzimanja matrica

4.2.6. Skalarno množenje

Fragment koji pruža skalarno množenje matrice prikazan je na slici 4.17. Sastoji se od ranije spomenutih *grid layouta*, gumba *calculate*, 3 *ImageViewa* te jednog *EditTexta* koji služi za unos broja s kojim množimo matricu, odnosno skalara. Funkcija za množenje skalarom prikazana je na slici 4.18.



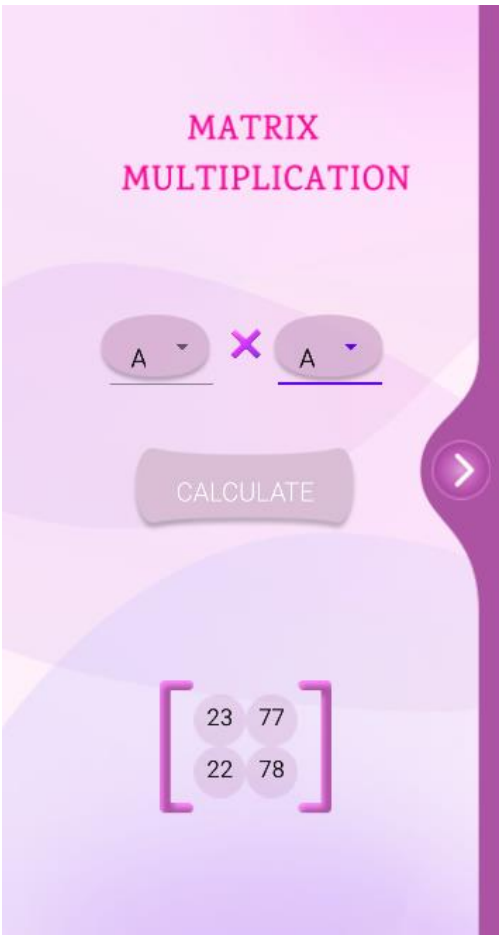
Sl. 4.17. Skalarno množenje matrice

```kotlin
fun multiplyMatrixByNumber(matrix: List<List<Int>>, number: Int): List<List<Int>> {
    val result = mutableListOf<MutableList<Int>>()

    for (row in matrix) {
        val newRow = mutableListOf<Int>()
        for (element in row) {
            newRow.add(element * number)
        }
        result.add(newRow)
    }

    return result
}
```

Sl. 4.18. Funkcija za množenje matrice skalarom

4.2.7. Množenje matrica

Fragment za množenje dviju matrica prikazan je na slici 4.19. Sastoji se od istih elemenata kao i fragment za zbrajanje i oduzimanje matrica. Jedina razlika je što se klikom na gumb *calculate* kao rezultat daje nova matrica dobivena množenjem dvije odabrane matrice. Funkcija za množenje matrica prikazana je na slici 4.20.



Sl. 4.19. Množenje dviju matrica

```
fun multiplyMatrices(matrix1: List<List<Int>>, matrix2: List<List<Int>>): List<List<Int>> {
    val numRows1 = matrix1.size
    val numCols1 = matrix1[0].size
    val numRows2 = matrix2.size
    val numCols2 = matrix2[0].size

    val result = MutableList(numRows1) { MutableList(numCols2) { 0 } }

    for (i in 0 ≤ until < numRows1) {
        for (j in 0 ≤ until < numCols2) {
            for (k in 0 ≤ until < numCols1) {
                result[i][j] += matrix1[i][k] * matrix2[k][j]
            }
        }
    }

    return result
```

Sl. 4.20. Funkcija za množenje dvije matrica

4.2.8. Računanje determinante

Fragment za izračun determinante nalazi se na slici 4.21. Sastoji se od gumba calculate, jednog *ImageViewa*, već spomenutog padajućeg izbornika sa odabirom 4 moguće matrice te jednog *TextViewa*, odnosno elementa XML-a koji omogućuje prikaz teksta, a u ovom slučaju prikazuje rezultat, odnosno kolika je determinanta. Računanje determinante prikazano je na slici 4.22.

Sl. 4.21. Računanje determinante matrice

```kotlin
private fun determinant(matrix: List<List<Int>>): Int {
    val size = matrix.size
    if (size == 1) {
        return matrix[0][0]
    }
    if (size == 2) {
        return matrix[0][0] * matrix[1][1] - matrix[0][1] * matrix[1][0]
    }
    var det = 0

    for (col in 0 ≤ until < size) {
        val cofactor = matrix[0][col] * determinant(getSubmatrix(matrix, rowToRemove: 0, col))
        det += if (col % 2 == 0) cofactor else -cofactor
    }

    return det
}

private fun getSubmatrix(matrix: List<List<Int>>, rowToRemove: Int, colToRemove: Int): List<List<Int>> {
    return matrix.filterIndexed { rowIndex, _ -> rowIndex != rowToRemove }
        .map { row -> row.filterIndexed { colIndex, _ -> colIndex != colToRemove } }
}
```

Sl. 4.22. Funkcija za računanje determinante matrice

5. ZAKLJUČAK

Aplikacija izrađena u okviru ovog rada namijenjena je za učenike, studente i ostale korisnike kako bi mogli olakšano provjeravati rezultate prilikom učenja i rješavanja matematičkih operacija s matricama. Korisniku se nudi unos dimenzije matrica sve do 4x4 te 5 matematičkih operacija s matricama: zbrajanje, oduzimanje, množenje skalarom, množenje dviju matrica te računanje determinante matrice. Naglasak ove aplikacije više je na estetici i vizualnom izgledu kako bi bila interaktivna za korištenje i privlačna nego na programskom dijelu, odnosno što većem mogućem broju matematičkih operacija. Cilj je privući što veći broj korisnika modernim sučeljem te uljepšati iskustvo korištenja. Kroz cijeli rad objašnjeno je koji su sve alati i znanja bili potrebni za izradu takve aplikacije.

LITERATURA

[1] MATRICE, Veleučilište u Požegi, dostupno na:

https://www.vup.hr/_Data/Files/12112291237379.pdf [22.6.2023.]

[2] Mibbmemima, Kako mogu znati da li je matrica degenerirana?, dostupno na:

https://mibbmemima.com/bs/como-puedo-saber-si-una-matriz-es-degenerada/ [22.6.2023.]

[3] Hrvatska enciklopedija, matrica, dostupno na:

https://www.enciklopedija.hr/natuknica.aspx?id=39492 [22.6.2023.]

[4] Hrvatska enciklopedija, determinanta, dostupno na:

https://enciklopedija.hr/natuknica.aspx?id=14816 [22.6.2023.]

[5] Hrvatsko strukovno nazivlje, inverzna matrica, dostupno na:

http://struna.ihjj.hr/naziv/inverzna-matrica/32321/ [22.6.2023.]

[6] Android pomoć, dostupno na: https://androidayuda.com/hr/android-studio/ [6.9.2023.]

[7] Wikipedija, XML, dostupno na: XML – Wikipedija (wikipedia.org) [6.9.2023.]

SAŽETAK

U ovom radu objašnjen je koncept izrađene aplikacije, odnosno mogućnost računanja različitih matematičkih operacija s matricama. Prikazana su već postojeća rješenja sličnih aplikacija te istaknute su prednosti i mane istih. Objašnjene su matrice, matematičke operacije s matricama te osnovni pojmovi koji se vežu uz matrice. Prikazano je programsko rješenje izrade aplikacije. Navedeni su alati koji su se koristili za izradu te objašnjeni osnovni pojmovi. Uključen je izgled aplikacije te slike programskog koda osnovnih funkcija.

Ključne riječi : Android Studio, aplikacija, matrice, XML

ABSTRACT

This paper explains the concept of the created application, the ability of calculating various mathematical operations with matrices. Already existing solutions of similar applications are presented, and the advantages and disadvantages are highlighted. Matrices, mathematical operations with matrices and basic terms related to matrices are explained. The software solution for creating the application is presented. The tools that were used for the creation are listed and the basic terms are explained. The appearance of the application and images of the program code of the basic functions are included.

Keywords: Android Studio, application, matrix, XML