ŠOLSKI CENTER VELENJE
ELEKTRO IN RAČUNALNIŠKA ŠOLA VELENJE
TRG MLADOSTI 3, 3320 VELENJE

MLADI RAZISKOVALCI ZA RAZVOJ SAŠA REGIJE

RAZISKOVALNA NALOGA

STEGANOGRAFIJA

Tematsko področje: RAČUNALNIŠTVO

AVTORJA:
Lara Koren
Tomaž Špegelj

MENTOR:
Islam Mušić, prof.

Velenje, 2026

Raziskovalna naloge je bila opravljena na Elektro in računalniški šoli Velenje, 2026.

Mentor: Islam Mušić, prof.

Datum predstavitve: marec 2026

KLJUČNA DOKUMENTACIJSKA INFORMACIJA

ŠD Elektro in računalniška šola Velenje, šolsko leto 2025/2026

KD program / programiranje / skrivanje podatkov / neopaznost

AV KOREN, Lara / ŠPEGELJ, Tomaž

SA MUŠIĆ, Islam

KZ 3320 Velenje, Trg mladosti 3

ZA ŠCV, Elektro in računalniška šola Velenje

LI 2026

IN **STEGANOGRAFIJA**

TD Raziskovalna naloga

OP VIII, 26 str., 8. vir., 2 pril.

IJ SL

JI sl / en

AI Raziskovalna naloga obravnava steganografijo kot metodo skrivanja informacij znotraj digitalnih datotek, pri čemer je poudarek na prikrivanju obstoja sporočila. Namen naloge je bil raziskati delovanje steganografije in razviti program, ki omogoča skrivanje in branje besedilnih sporočil.

V okviru naloge je bil razvit program v programskem jeziku C#, ki omogoča steganografijo v slikovnih in tekstovnih datotekah. Pri skrivanju podatkov v slikah je uporabljena metoda najmanj pomembnega bita (LSB), pri kateri program glede na dolžino sporočila samodejno določi število uporabljenih bitov, s čimer zagotovi optimalno razmerje med kapaciteto in neopaznostjo sprememb slike.

Cilj raziskovalne naloge je prikazati praktično uporabo steganografije ter njeno primernost za diskretno shranjevanje in prenos informacij v digitalnem okolju.

KEY WORDS DOCUMENTATION

ND Electrical and Computer School Velenje, school year 2025/2026

CX program / programming / data hiding / inconspicuousness

AU KOREN, Lara / ŠPEGELJ, Tomaž

AA MUŠIĆ, Islam

PP 3320 Velenje, SLO, Trg mladosti 3

PB ŠCV, Electrical and Computer School Velenje

PY 2026

TI **STEGANOGRAPHY**

DT Research work

NO VIII, 26 p., 8. vir., 2 ann.

LA SL

AL sl / en

AB The research paper deals with steganography as a method of hiding information within digital files, with the emphasis on concealing the existence of a message. The purpose of the paper was to investigate the operation of steganography and develop a program that enables the hiding and reading of text messages.

As part of the paper, a program was developed in the C# programming language that enables steganography in image and text files. The least significant bit (LSB) method is used to hide data in images, in which the program automatically determines the number of bits used based on the length of the message, thus ensuring the optimal ratio between capacity and invisibility of image changes.

The aim of the paper is to demonstrate the practical use of steganography and its suitability for discrete storage and transmission of information in a digital environment

KAZALO VSEBINE

KAZALO SLIK

UPORABLJENI SIMBOLI

#K# – zaključna oznaka za konec skritega sporočila

B – modri barvni kanal piksla

G – zeleni barvni kanal piksla

LSB – najmanj pomemben bit pri zapisu podatkov (angl. Least Significant Bit)

R – rdeči barvni kanal piksla

RGB – barvni model, sestavljen iz kanalov rdeča, zelena in modra

UPORABLJENE KRATICE

.NET – programsko ogrodje za razvoj aplikacij

API – vmesnik za programiranje aplikacij (angl. Application Programming Interface)

ASCII – standard za zapis znakov s številskimi vrednostmi

BMP – slikovni format Bitmap

DCT – diskretna kosinusna transformacija (angl. Discrete Cosine Transform)

DWT – diskretna valčna transformacija (angl. Discrete Wavelet Transform)

ERŠ – Elektro in računalniška šola (ERŠ)

FFT – hitra Fourierova transformacija (angl. Fast Fourier Transform)

GUI – grafični uporabniški vmesnik

HAS – človeški slušni sistem (angl. Human Auditory System)

IDE – integrirano razvojno okolje

LPC – linearno napovedno kodiranje (angl. Linear Predictive Coding)

MP3 – format za stiskanje zvočnih datotek

PNG – slikovni format (angl. Portable Network Graphics)

ŠCV – Šolski center Velenje (Šolski center Velenje)

VR – navidezna resničnost (angl. Virtual Reality)

WPF – Windows Presentation Foundation (tehnologija za izdelavo GUI aplikacij v .NET)

1 UVOD

1.1 OPIS PODROČJA IN PROBLEM

Iz dneva v dan se število internetnih uporabnikov hitro povečuje, s tem eksponentno narašča tudi količina podatkov, ki se prenašajo po različnih omrežjih. Posledično se povečujejo tudi varnostna tveganja, zato postajajo varnost, zasebnost in zaščita vse pomembnejša področja sodobne informacijske družbe.

Klasične metode kot na primer šifriranje, so učinkovite za zaščito vsebine podatkov ali sporočil, vendar ne prikrijejo dejstva, da komunikacija poteka. Takšna sporočila so pogosto tarča napadalcev ali nadzornih sistemov.

Za zaščito podatkov obstajajo tudi drugi, manj opazni pristopi. Eden izmed teh je steganografija, ki temelji na skrivanju sporočila znotraj drugih digitalnih nosilcev, najpogosteje so to slike, zvočne datoteke ali besedilo.

1.2 NAMEN IN CILJI

Namen te raziskovalne naloge je predstaviti osnovne koncepte steganografije in kako je uporabljena v sodobnem digitalnem okolju. Poseben poudarek sva dala na skrivanje podatkov v slikovne datoteke z uporabo metode najmanj pomembnega bita (angl. LSB).

Cilj naloge je razviti program, ki omogoča vstavljanje in razbiranje tekstovnega sporočila iz digitalnih slikovnih datotek. Program bo pri vstavljanje teksta v slike prilagodljiv na dolžino sporočila in velikosti slike za najmanj opažne sledi sporočila.

1.3 METODE DELA

Pri izdelavi raziskovalne naloge bova uporabila metodo pregleda že obstoječe literature s področja steganografije, analizo obstoječih programskih rešitev ter praktično delo v obliki načrtovanja, programiranja in testiranja razvitega programa. Končni program bova tudi analizirala in testirala.

1.4 HIPOTEZE

1.4.1 PRVA HIPOTEZA

Steganografija v slikovnih datotekah omogoča skrivanje podatkov na način, ki je za človeško oko neopažen.

1.4.2 DRUGA HIPOTEZA

Razvit program omogoča preprosto in uporabniku prijazno skrivanje podatkov v slikovne datoteke.

1.4.3 TRETJA HIPOTEZA

Velikost slike in dolžina skritega sporočila vplivata na stopnjo neopaznosti pri steganografiji v slikovnih datotekah.

2 TEORETIČNI DEL

2.1 STEGANOGRAFIJA

Steganografija je postopek skrivanja tajnega sporočila tako, da ni očitno, da komunikacija sploh poteka. Glavni namen steganografije ni zakrivanje vsebine sporočila, temveč prikrivanje njegovega obstoja.

Pri steganografiji se tajno sporočilo vstavi v drugo, povsem običajno datoteko, ki se imenuje nosilec. Nosilec je lahko besedilo, slika, zvočni ali video posnetek, saj so to vrste podatkov, ki se pogosto uporabljajo v vsakdanji komunikaciji in zato ne vzbujajo suma.

Datoteka, ki vsebuje tako nosilec kot tudi vanj vstavljeno tajno sporočilo, se imenuje steganografski medij oziroma stego. Za dodatno varnost se lahko uporabi tudi steganografski ključ, s katerim se tajno sporočilo pred vstavljanjem še dodatno zaščiti.

Steganografski medij lahko torej opišemo kot kombinacijo nosilca, tajnega sporočila in po potrebi steganografskega ključa.

2.1.1 SPLOŠEN OPIS STEGANOGRAFIJE

Steganografija se v digitalnem svetu najpogosteje uporablja pri slikah, zvočnih datotekah in video posnetkih. Ker so takšne datoteke zelo razširjene v vsakdanji rabi, so primerne za skrivanje podatkov, saj ne vzbujajo posebne pozornosti. Skrito sporočilo je vdelano tako, da spremembe niso zaznavne s prostim očesom ali sluhom.

Pri digitalnih slikah se steganografija pogosto izvaja z manjšimi spremembami barvnih vrednosti posameznih slikovnih pik. Te spremembe so za uporabnika neopazne, vendar omogočajo shranjevanje dodatnih podatkov v sliko. Podoben princip se uporablja tudi pri zvočnih in video datotekah, kjer se podatki skrivajo v najmanj opaznih delih signala.

V praktičnem delu raziskovalne naloge je ta način skrivanja podatkov prikazan s pomočjo računalniškega programa, ki omogoča vdelavo in razkrivanje skritega besedila v digitalni sliki. [4]

2.1.2 STEGANOGRAFIJA V TEKSTOVNIH DATOTEKAH

Besedilna steganografija je zahtevna tehnika, saj tekstovne datoteke vsebujejo zelo malo odvečnih podatkov, v primerjavi z drugimi digitalnimi mediji, kot so slike, zvok ali video. Zaradi tega je skrivanje informacij v besedilu težje izvedljivo. Besedilno steganografijo lahko na splošno razdelimo na tri glavne vrste:

1. **Jezikovna steganografija** – uporablja spremembe besed, slovnice ali stavčnih struktur za kodiranje skritih informacij.
2. **Formatna steganografija** – temelji na spremembah fizičnih značilnosti besedila, na primer razmikih med besedami, premikanju vrstic ali postavitvi simbolov.
3. **Naključna in statistična steganografija** – skrito sporočilo se vnaša z uporabo naključnih ali statističnih metod, ki so manj opazne za bralca. [6]

2.1.3 STEGANOGRAFIJA V SLIKOVNIH DATOTEKAH

Slikovna steganografija je tehnika skrivanja informacij znotraj digitalnih slik, pri čemer se vizualni videz slike praktično ne spremeni. Namen te metode je omogočiti tajno komunikacijo, saj lahko občutljive informacije prenesemo, ne da bi vzbudili sum pri nepooblaščenih osebah ali nasprotnikih.

Pri digitalnih slikah se informacije pogosto vgrajujejo v najmanj pomembne bite (angl. LSB) posameznih slikovnih pik. Spremembe so za človeško oko skoraj neopazne, kar omogoča diskretno prenašanje podatkov. Poleg tega slikovna steganografija ponuja dodatno plast varnosti in zaupnosti v digitalni komunikaciji, saj je prisotnost skritih podatkov sama po sebi neopazna. [7]

2.1.4 STEGANOGRAFIJA V VIDEO DATOTEKAH

Steganografija v video datotekah je razširitev slikovne steganografije, pri kateri se sporočilo skriva znotraj digitalnega video posnetka. Video je sestavljen iz zaporedja slikovnih okvirjev, kar omogoča uporabo podobnih tehnik kot pri slikah, pri čemer se vizualna kakovost videa bistveno ne spremeni.

Informacije se najpogosteje vgrajujejo v posamezne okvirje videa, kjer se spreminjajo najmanj pomembni biti (angl. LSB) barvnih vrednosti slikovnih pik. Zaradi velike količine podatkov in manjše občutljivosti človeškega vida na majhne spremembe med zaporednimi sličicami so takšne spremembe za gledalca skoraj neopazne, kar omogoča diskretno prenašanje skritih informacij. [8]

2.1.5 STEGANOGRAFIJA V ZVOČNIH DATOTEKAH

Zvočna steganografija je tehnika skrivanja podatkov znotraj zvočnih ali govornih signalov, pri čemer se izvorni zvok s prostim ušesom skoraj ne spremeni. Namen je omogočiti varen prenos informacij, saj je skrito sporočilo neopazno za poslušalce in nepooblaščene osebe.

Metode zvočne steganografije se lahko razvrstijo glede na princip skrivanja podatkov. Med najpogostejšimi so:

- **HAS (angl. Human Auditory System) tehnike** - izkoriščajo značilnosti človeškega sluha, da spremembe zvoka ostanejo neopazne.
- **LSB (angl. Least Significant Bit) metoda** - spremeni najmanj pomembne bite zvočnih vzorcev za vgradnjo sporočila.
- **LPC (angl. Linear Predictive Coding)**, **DWT (angl. Wavelet transformations)**, **DCT (angl. Discreet Cosine Transform)** - uporabljajo transformacije signalov za vgradnjo podatkov.
- **Fazno kodiranje (angl. Phase coding), skrivanje odmeva (echo hiding), Razpršeni spekter (angl. Spread spectrum)** - naprednejše metode, ki manipulirajo fazo, odmev ali frekvenčne lastnosti zvoka.
- **Skrivanje podatkov MP3 (angl. MP3 data hiding)** - metode za skrivanje podatkov v stisnjenih MP3 datotekah.

Metode se glede na obdelavo signala delijo tudi na:

1. **Temporalno domeno** – neposredna sprememba zvočnih vzorcev, kot so LSB substitucija, skrivanje odmeva in časovno maskiranje.
2. **Transformno domeno** – sprememba zvoka v transformirano obliko (FFT, DCT, DWT) in vgradnja sporočila v koeficiente transformacije, na primer z uporabo frekvenčnega maskiranja ali fazno kodiranih tehnik.

2.1.6 METODA LSB

Ta metoda je ena izmed bolj pogosto uporabljenih tehnik steganografije, predvsem pri slikovnih datotekah. Deluje tako, da spremeni najmanj pomemben bit (angl. LSB – Least Significant Bit) posamezne barvne komponente piksla, pri čemer sprememba do določene mere ne povzroči opazne razlike v videzu slike.

Za boljše razumevanje je potrebno vedeti, kako so piksli sestavljeni. Vsak piksel vsebuje tri barvne kanale: **R**ed, **G**reen in **B**lue (**RGB**), kjer ima vsak kanal vrednost med 0 in 255. Ker je vsaka barvna komponenta zapisana z 8 biti, lahko pri metodi LSB spreminjamo samo zadnji bit zapisa, kar povzroči zelo majhno spremembo barve, ki je človeško oko večinoma ne zazna (npr. sprememba iz 255 na 254).

Skrivanje sporočila poteka tako, da se biti besedila shranjujejo v najmanj pomembne bite kanalov RGB. Ker ima vsak piksel tri kanale, lahko v enem pikslu shranimo 3 bite. Besedilo najprej pretvorimo v ASCII vrednosti in nato v binarno obliko, na primer črka “a” ima vrednost 97 oziroma binarno 01100001. Pri branju nato iz slike ponovno preberemo najmanj pomembne bite, jih združimo v 8-bitne skupine in jih pretvorimo nazaj v znake, s čimer dobimo originalno sporočilo. [2]

2.1.7 STEGANALIZA

Je veda, ki se ukvarja z odkrivanjem podatkov in sporočil, ki so bila z postopkom steganografije skrita v različne digitalne nosilce. Medtem ko je steganografija namenjena skrivanju podatkov, je steganaliza njegovo nasprotje, saj je njen cilj ugotoviti, ali datoteka vsebuje skrito sporočilo.

Pri steganalizi se uporabljajo različne metode, kot so statična analiza podatkov, analiza histogramov slik in zaznavanje nepravilnosti v strukturi digitalnih nosilcev. Čeprav so spremembe pri metodi LSB za človeško oko skoraj neopazne, jih z naprednimi postopki mogoče zaznati.

2.2 UPORABLJENA TEHNOLOGIJA IN ORODJA

2.2.1 C#

C# (izgovorjava C-Sharp) je objektno usmerjen programski jezik, ki ga je razvilo podjetje Microsoft in deluje na ogrodju .NET. Temelji na družini jezikov C in je po strukturi podoben jezikoma C++ in Java. Prva različica je bila izdana leta 2002, najnovejša pa je C# 14, ki je izšla novembra 2025.

C# se uporablja za razvoj različnih vrst programov, kot so namizne, spletne in mobilne aplikacije, spletne storitve, igre, VR ter aplikacije za delo z bazami podatkov. Je zelo priljubljen, ker je enostaven za učenje, ima veliko podporo skupnosti in omogoča pregledno ter ponovno uporabno kodo, kar olajša razvoj in zmanjšuje stroške. [1]

2.2.2 VISUAL STUDIO

Visual Studio je integrirano razvojno okolje (angl. IDE), ki ga je razvilo podjetje Microsoft. Uporablja se za razvoj različnih programov, kot so namizne aplikacije, spletne strani, spletne storitve in mobilne aplikacije. Podpira Microsoftove razvojne platforme, kot so Windows API, Windows Forms in WPF.

Vsebuje urejevalnik kode z IntelliSense (samodejno dopolnjevanje), orodja za preoblikovanje kode, vgrajen razhroščevalnik (angl. debugger) ter različne dodatne pripomočke, kot so oblikovalnik uporabniškega vmesnika, profiler in orodja za delo z bazami. Visual Studio podpira veliko programskih jezikov (npr. C#, C++, JavaScript, HTML, CSS), dodatne jezike pa je mogoče dodati z razširitvami. Na voljo je v več različicah, pri čemer je Community brezplačna za študente in posameznike.

2.2.3 WINDOWS FORMS

Windows Forms (pogosto skrajšano WinForms) je tehnologija za izdelavo grafičnih uporabniških vmesnikov (GUI) v programskem jeziku C# in .NET Framework. Omogoča enostavno ustvarjanje aplikacij za operacijski sistem Windows, ki imajo okna, gumbe, besedilna polja, menije in druge vizualne elemente.

2.2.4 .NET

.NET je ogrodje, ki ga je izdelal Microsoft za razvijanje programske opreme, ki teče večinoma na operacijskih sistemih Microsoft Windows.

2.2.5 BITMAP

Bitmap je oblika digitalne slike, ki jo sestavljajo posamezne slikovne pike (angl. pixels), pri čemer vsaka pika vsebuje informacije o barvi. Vsaka slikovna pika ima običajno tri barvne komponente: rdečo (R), zeleno (G) in modro (B), kar omogoča prikaz širokega spektra barv.

V programskem jeziku C# je razred Bitmap del knjižnice System.Drawing in omogoča enostaven dostop do posameznih slikovnih pik. Z razredom Bitmap lahko program:

- bere in spreminja vrednosti posameznih pik,
- ustvarja nove slike,
- shranjuje slike v različne formate,
- manipulira s slikami pri algoritmih steganografije.

Pri slikovni steganografiji je razred Bitmap še posebej uporaben, saj omogoča vdelavo ali razkritje skritih podatkov na nivoju posameznih slikovnih pik, ne da bi pri tem spremenil opazen videz slike. Z njegovo pomočjo lahko program prebere sliko, spremeni najmanj pomembne bite (angl. LSB) posameznih pik in nato shrani spremenjeno sliko kot novo datoteko.

3 RAZVOJ IN IZDELAVA

3.1 NAČRTOVANJA REŠITVE

Najina rešitev temelji na metodi LSB steganografije, kjer se sporočilo skriva v najmanj pomembne bite barvnih kanalov (angl. RGB) posameznih pikslov. Glavni cilj je bil izdelati program, ki omogoča skrivanje in branje sporočil iz slikovnih in video datotek, pri čemer program samodejno določi število uporabljenih LSB bitov (od 1 do 4) glede na dolžino sporočila ter ločljivost slike oziroma videa. Na ta način se zagotovi zadostna kapaciteta za shranjevanje sporočila ob hkrati čim manjši vidni spremembi slike ali videa.

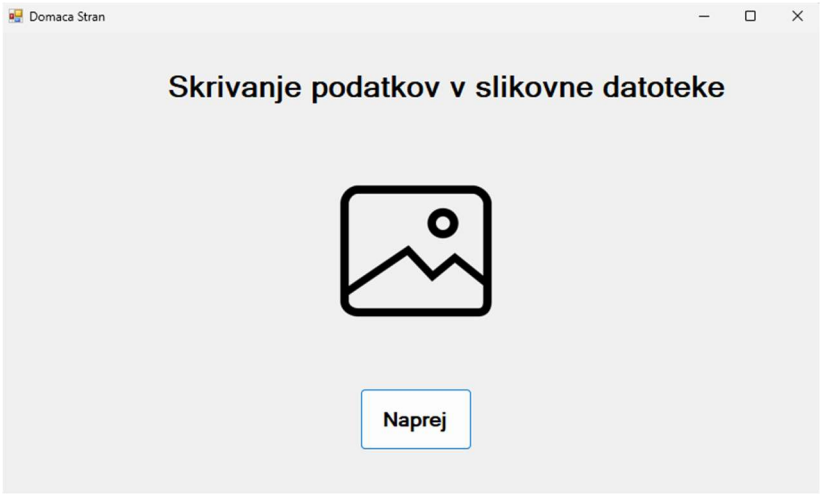
Pri načrtovanju sva se osredotočila na to, da program najprej zapolni prvi LSB vseh pikslov, nato drugi LSB, nato tretji in po potrebi četrti LSB. Tak način zapisa zmanjša opazne spremembe, saj se najprej spreminjajo najmanj pomembni biti, šele nato bolj izraziti biti. Za lažje branje sporočila sva na konec besedila dodala oznako »#K#«, ki programu pove, kdaj se sporočilo konča, zato dolžine sporočila ni potrebno poznati vnaprej. Rešitev je sestavljena iz dveh delov: skrivanja sporočila v slikovne oziroma video datoteke ter branja sporočila iz njih, pri čemer program deluje na slikah formata PNG in BMP ter na video datotekah.

3.2 OPIS POSTOPKA - STEGANOGRAFIJA SLIKE

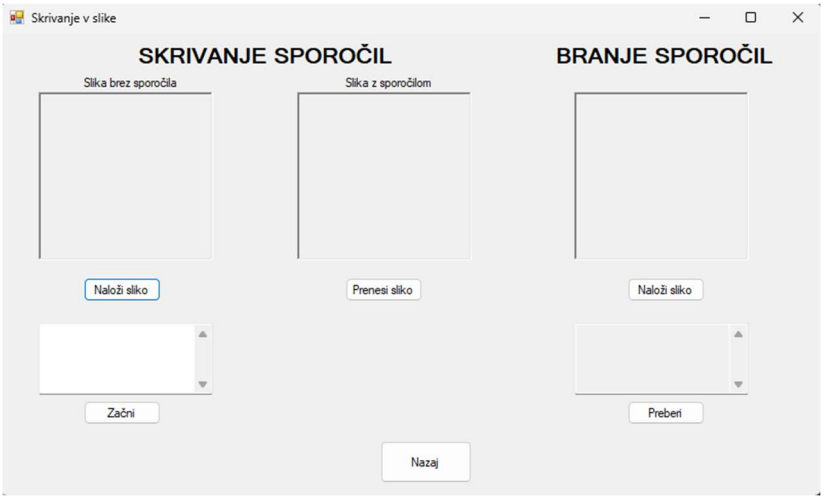
Izdelavo programa sva začela z raziskovanjem steganografije, kaj je, kako deluje in kako lahko narediva najin program, ki ima implementirano steganografijo. Ko sva se dovolj informirala v tej temi, sva se odločila za programski jezik in orodje, ki ju bova uporabila (jezik C# v ogrodju .NET z uporabo Visual Studio).

Zdaj sva začela z dejansko izdelavo programa. Pri pregledu že obstoječe literature sva odkrila nekaterih že obstoječih programov, ki so istega oziroma podobnega koncepta, kot najin (povezave do teh programov so navedene v poglavju viri in literatura).

V razvojnem okolju Visual Studio sva začela z izdelavo uporabniškega vmesnika, ki je izgledal kot je prikazano na spodnjih slikah.

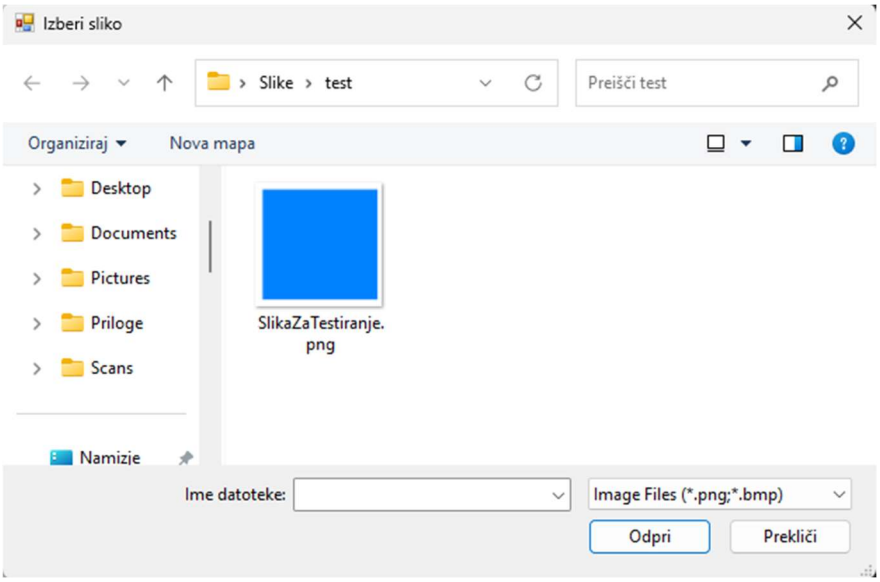


Slika 1 Izgled uporabniškega vmesnika programa (Domača stran)

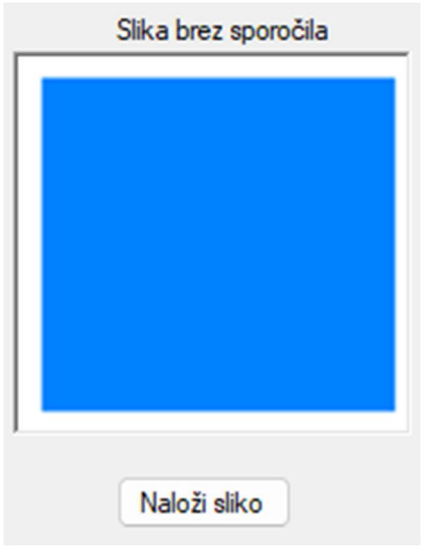


Slika 2 Izgled uporabniškega vmesnika programa (Skrivanje v slike)

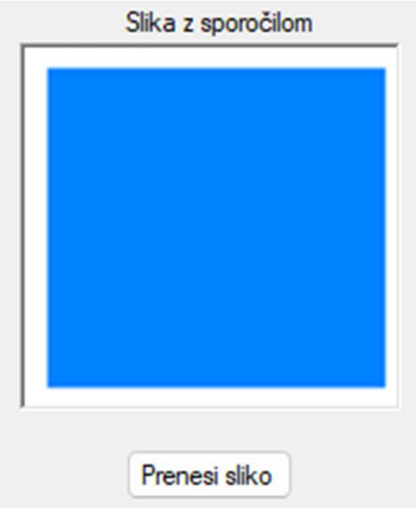
Nato sva implementirala možnost nalaganja slik v program. Uporabila sva OpenFileDialog, ki uporabniku omogoča izbiro slike v formatu .png ali .bmp. V programu sva naredila dve možnosti nalaganja: ena slika je namenjena skrivanju sporočila (naloziSkrivanjeButton), druga pa je namenjena branju sporočila (naloziBranjeButton). Ko uporabnik izbere sliko, se ta prikaže v ustreznem PictureBox elementu (predPictureBox za skrivanje in branjePictureBox za branje). Pri tem sva uporabila tudi začasni Bitmap temp, da se slika ne zaklene in jo lahko kasneje brez težav shranjujeva ali ponovno odpirava.



Slika 3 Izbira slike preko okna OpenFileDialog



Slika 4 Naložena slika za skrivanje sporočila (prikaz v predPictureBox)



Slika 5 Naložena slika za branje sporočila (prikaz v branjePictureBox)

Ko je bila slika pravilno naložena, sva implementirala pretvorbo sporočila v binarno obliko. Za to sva napisala funkcijo VBinarnoOblikoChar, ki vsak znak iz sporočila pretvori v 8-bitni binarni zapis. Program nato združi vse bite v en dolg niz (binSporocilo), ki ga kasneje uporabi pri skrivanju sporočila v piksle slike. [3]

```
1 reference
static string VBinarnoOblikoChar(char x)
{
    int zacasno = Convert.ToInt32(x);
    string binarno = "";

    while (zacasno > 0)
    {
        int ostanek = zacasno % 2;
        binarno = ostanek + binarno;
        zacasno /= 2;
    }

    for (int i = binarno.Length; i < 8; i++)
    {
        binarno = "0" + binarno;
    }

    return binarno;
}
```

Slika 6 Del kode za pretvorbo znakov v binarno obliko (VBinarnoOblikoChar)

Za skrivanje sporočila v sliko sva uporabila metodo LSB (Least Significant Bit). To pomeni, da se bit sporočila shrani v najmanj pomemben bit barvne vrednosti piksla. V programu sva to izvedla s funkcijo NastaviLSB, ki spremeni izbran bit v vrednosti barvnega kanala (R, G ali B). S tem doseževa, da se slika vizualno skoraj ne spremeni, sporočilo pa je vseeno zapisano v njenih barvah.

```
3 references
static int NastaviLSB(int barva, char bit, int indeks)
{

    int pozicija = 7 - indeks;

    string binarnaBarva = Convert.ToString(barva, 2).PadLeft(8, '0');

    StringBuilder novaBarva = new StringBuilder(binarnaBarva);
    novaBarva[pozicija] = bit;

    return Convert.ToInt32(novaBarva.ToString(), 2);
}
```

Slika 7 Del kode za nastavljanje bitov v barvni kanal (NastaviLSB)

Nato sva napisala funkcijo SkrijSporocilo, ki gre skozi piksle slike in v njihove barvne kanale zapisuje bite sporočila. Program zapisuje bite v vrstnem redu po pikslih in barvah (R, G, B). Poleg tega sva implementirala tudi možnost, da program po potrebi uporabi več bitov (od 1 do 4 LSB). To deluje tako, da najprej zapolni 1. LSB pri vseh pikslih, nato 2. LSB, nato 3. in po potrebi še 4. LSB. Na ta način se v sliko lahko skrije daljše sporočilo, ne da bi takoj uporabili preveč bitov in s tem pokvarili kvaliteto slike.

```
1 reference
static Bitmap SkrijSporocilo(Bitmap slika, string binSporocilo, int maxLSB)
{
    int indeks = 0;

    for (int bitIndex = 0; bitIndex < maxLSB; bitIndex++)
    {
        for (int y = 0; y < slika.Height; y++)
        {
            for (int x = 0; x < slika.Width; x++)
            {

                if (indeks >= binSporocilo.Length)
                {
                    return slika;
                }

                Color piksel = slika.GetPixel(x, y);

                int r = piksel.R;
                int g = piksel.G;
                int b = piksel.B;

                if (indeks < binSporocilo.Length)
                {
                    r = NastaviLSB(r, binSporocilo[indeks++], bitIndex + 1);
                }

                if (indeks < binSporocilo.Length)
                {
                    g = NastaviLSB(g, binSporocilo[indeks++], bitIndex + 1);
                }

                if (indeks < binSporocilo.Length)
                {
                    b = NastaviLSB(b, binSporocilo[indeks++], bitIndex + 1);
                }

                slika.SetPixel(x, y, Color.FromArgb(r, g, b));
            }
        }
    }
```

Slika 8 Del kode za skrivanje sporočila v sliko (SkrijSporocilo)

Da program ve, kdaj se sporočilo konča, sva dodala poseben zaključni niz #K#. Program ta niz avtomatsko doda na konec sporočila pred skrivanjem (sporocilo += "#K#";). Pri branju se nato preberejo znaki iz slike, dokler se ne pojavi #K#, potem pa program vrne samo originalno sporočilo brez zaključka.

```
if (rezultat.ToString().EndsWith("#K#"))
    return rezultat.ToString().Replace("#K#", "");
```

Slika 9 Preverjanje končnega znaka "#K#" pri branju sporočila

Preden se sporočilo skrije, program preveri, ali je sporočilo sploh možno shraniti v izbrano sliko. Izračuna se največje možno število znakov glede na velikost slike in število uporabljenih LSB bitov (1–4). Če je sporočilo preveliko, program uporabniku izpiše obvestilo “Sporočilo je predolgo!”. Na podlagi dolžine sporočila se avtomatsko nastavi vrednost lsb, ki določa koliko bitov bo program uporabil.

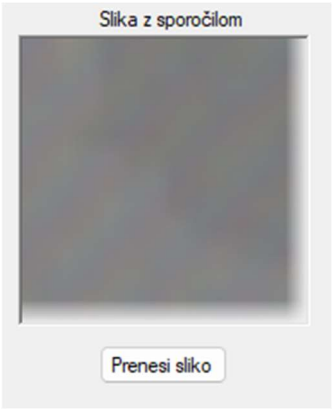
```
int limitEna = ((novaSlika.Height * novaSlika.Width) * 3) / 8;
int limitDva = ((novaSlika.Height * novaSlika.Width) * 6) / 8;
int limitTri = ((novaSlika.Height * novaSlika.Width) * 9) / 8;
int limitStiri = ((novaSlika.Height * novaSlika.Width) * 12) / 8;
```

Slika 10 Del kode za izračun omejitev

```
if (sporocilo.Length <= limitEna)
{
    lsb = 1;
}
else if (sporocilo.Length <= limitDva)
{
    lsb = 2;
}
else if (sporocilo.Length <= limitTri)
{
    lsb = 3;
}
else if (sporocilo.Length <= limitStiri)
{
    lsb = 4;
}
else
{
    MessageBox.Show("Sporočilo je predolgo!");
    return;
}
```

Slika 11 Del kode za nastavitev število uporabljenih lsb

Ko je sporočilo skrito, se nova slika prikaže v drugi slikovni škatli, da uporabnik takoj vidi rezultat. Program nato omogoča shranjevanje slike z gumbom, ki sliko shrani kot SkritaSlika.png. Pri tem sva upoštevala tudi primer, da taka datoteka že obstaja, zato program shrani sliko pod imenom SkritaSlika2.png, SkritaSlika3.png, itd., da ne prepiše stare datoteke.



Slika 12 Prikaz slike z sporočilom

Ime	Datum spremembe	Vrsta	Velikost
SkritaSlika.png	24. 01. 2026 16:38	Datoteka PNG	1 KB
SkritaSlika2.png	24. 01. 2026 16:38	Datoteka PNG	1 KB
SkritaSlika3.png	24. 01. 2026 16:38	Datoteka PNG	1 KB

Slika 13 Primer shranjevanja več slik

Za branje skritega sporočila sva implementirala funkciji PreberiLSB in PreberiSporocilo. Program prebere LSB bite iz barvnih kanalov pikslov (R, G, B), jih združi v 8-bitne skupine in pretvori nazaj v znake. Branje poteka do trenutka, ko se zazna zaključni niz #K#. Ko uporabnik klikne gumb za branje (button4), program izpiše skrito sporočilo v MessageBox.

```
3 references
static char PreberiLSB(int barva, int indeks)
{
    int pozicija = 7 - indeks;
    string binarna = Convert.ToString(barva, 2).PadLeft(8, '0');
    return binarna[pozicija];
}
```

Slika 14 Del kode za branje bitov iz slike (PreberiLSB)

```
1 reference
static string PreberiSporocilo(Bitmap slika, int maxLSB)
{
    StringBuilder biti = new StringBuilder();
    StringBuilder rezultat = new StringBuilder();

    for (int bitIndex = 0; bitIndex < maxLSB; bitIndex++)
    {
        for (int y = 0; y < slika.Height; y++)
        {
            for (int x = 0; x < slika.Width; x++)
            {
                Color barvaPiksla = slika.GetPixel(x, y);

                biti.Append(PreberiLSB(barvaPiksla.R, bitIndex + 1));
                biti.Append(PreberiLSB(barvaPiksla.G, bitIndex + 1));
                biti.Append(PreberiLSB(barvaPiksla.B, bitIndex + 1));

                while (biti.Length >= 8)
                {
                    string enBit = biti.ToString(0, 8);
                    biti.Remove(0, 8);

                    char znak = (char)Convert.ToInt32(enBit, 2);
                    rezultat.Append(znak);

                    if (rezultat.ToString().EndsWith("#K#"))
                        return rezultat.ToString().Replace("#K#", "");
                }
            }
        }
    }

    return rezultat.ToString();
}
```

Slika 15 Del kode za branje sporočila iz slike (PreberiSporocilo)

4 ANALIZA

4.1 KONČNI IZDELEK

Izdelala sva program, ki omogoča skrivanje in branje besedilnih sporočil v slikovne datoteke. Program je bil razvit v razvojnem okolju Visual Studio in temelji na metodi LSB steganografije. Uporabniku omogoča nalaganje slik ter samodejno prilagodi način skrivanja glede na velikost nosilne datoteke in dolžino sporočila.

Končni izdelek zagotavlja enostavno uporabo ter minimalno vidne spremembe slik, hkrati pa omogoča zanesljivo pridobivanje skritega sporočila.

4.2 TESTIRANJE

4.2.1 SPLOŠNO TESTIRANJE

Po zaključku razvoja programa sva izvedla celovito testiranje funkcionalnosti. Pri tem sva preverila vse ključne funkcionalnosti: nalaganje slik, skrivanje sporočil, branje skritih sporočil ter shranjevanje novih slik. Testiranje je vključevalo različne formate slik (.png in .bmp) ter različne dolžine sporočil.

Med testiranjem sva preverila, ali program pravilno pretvori sporočilo v binarno obliko, ga zapiše v LSB pikslov in nato uspešno prebere nazaj. Prav tako sva testirala mehanizem avtomatske nastavitve števila uporabljenih LSB bitov glede na dolžino sporočila in velikost slike, da se zagotovi optimalno skrivanje sporočila brez vidne degradacije slike.

Vsa testiranja so pokazala, da program deluje stabilno, sporočila se natančno skrijejo in preberejo, datoteke pa se shranijo brez napak in prepisovanja obstoječih slik.

4.2.2 OPAŽNOST SPREMEMB SLIKE

Posebno pozornost sva namenila opažnosti sprememb v slikah po skrivanju sporočil. Testiranje je potekalo tako, da sva enako sliko prikazala različnim osebam in jih prosila, naj ocenijo, ali opazijo kakršnekoli vizualne spremembe.

Rezultati so pokazali, da so spremembe skoraj neopažene pri uporabi ene ali dveh LSB bitov, celo pri treh bitih so spremembe minimalne in jih povprečna oseba težko zazna. Šele pri uporabi štirih LSB bitov so bile spremembe nekoliko bolj vidne pri posameznih barvnih prehodih, vendar še vedno niso bistveno vplivale na kakovost slike.

Na podlagi teh testov sva potrdila, da metoda LSB omogoča učinkovito skrivanje sporočil z minimalnim vplivom na vizualno kakovost slike, kar zagotavlja zanesljivost in diskretnost končnega izdelka.

4.3 MOŽNE IZBOLJŠAVE

Program bi bilo v prihodnje še dodatno izboljšati in razširiti. Ena izmed bolj pomembnih izboljšav bi bila podpora večjemu številu slikovnih in video formatov. Prav tako bi bilo možno dodati funkcijo šifriranja sporočila pred samim skrivanjem, kar bi povečalo varnost skritih podatkov.

Kot razširitev bi programu dodala tudi možnost za uporabo več vrst steganografije, poleg slikovne. Torej da bi lahko skril sporočilo v video, tekstovno ali zvočno datoteko.

5 RAZPRAVA IN HIPOTEZE

5.1 RAZPRAVA

V raziskovalni nalogi sva teoretično in praktično preverila delovanje steganografije z metodo LSB. Rezultati testiranja so pokazali, da je pri uporabi enega ali dveh LSB bitov sprememba slike za človeško oko skoraj neopazna. Pri treh bitih so spremembe minimalne, pri štirih pa se lahko pri določenih slikah že rahlo opazijo.

Ugotovila sva, da velikost slike pomembno vpliva na kakovost rezultata. Večje slike omogočajo skrivanje daljših sporočil z manjšo izgubo kakovosti, medtem ko manjše slike zahtevajo uporabo več LSB bitov, kar poveča možnost vidnih sprememb.

Program deluje stabilno in uporabniku omogoča enostavno skrivanje ter branje sporočil. Omejitev predstavlja podpora le določenim formatom slik ter odsotnost dodatnega šifriranja sporočila.

Kljub temu lahko zaključiva, da metoda LSB omogoča učinkovito in relativno neopazno skrivanje podatkov v slikovne datoteke.

5.2 PREVERJANJE HIPOTEZ

Prvo hipotezo potrjujemo, saj so spremembe pri uporabi manjšega števila LSB bitov za človeško oko neopažene.

Drugo hipotezo potrjujemo, ker je razvit program pregleden, enostaven za uporabo in deluje brez napak.

Tretjo hipotezo potrjujemo, saj sta velikost slike in dolžina sporočila neposredno vplivali na stopnjo neopaznosti sprememb.

6 ZAKLJUČEK

V zaključni nalogi sva raziskala steganografijo in izdelala program, ki omogoča skrivanje ter branje sporočil v slikovne datoteke z uporabo metode LSB (angl. Least Significant Bit). Ugotovila sva, da je LSB učinkovita metoda, saj lahko v sliko shranimo besedilo brez opaznih sprememb videza, dokler uporabljamo manjše število bitov (npr. 1 ali 2 LSB). Program sva razvila v jeziku C# v okolju Visual Studio, kjer sva izdelala uporabniški vmesnik, ki omogoča enostavno nalaganje slik, skrivanje sporočil ter njihovo branje.

Pri analizi rezultatov sva opazila, da večje število uporabljenih LSB bitov (3 ali 4) omogoča shranjevanje daljših sporočil, vendar se pri tem kakovost slike lahko bolj poslabša, kar postane opazno predvsem pri slikah z manj detajli ali z enakomernimi barvnimi površinami. Za pravilno branje sporočila sva uporabila oznako "#K#", ki označuje konec skritega besedila, kar se je izkazalo kot praktična rešitev, saj ni potrebno posebej shranjevati dolžine sporočila.

Na podlagi rezultatov lahko prvo hipotezo potrdimo, saj se je izkazalo, da steganografija v slikovnih datotekah omogoča skrivanje podatkov na način, ki je za človeško oko pri uporabi manjšega števila LSB bitov neopažen. Prav tako lahko potrdimo drugo hipotezo, saj razvit program omogoča preprosto in uporabniku prijazno skrivanje ter branje podatkov. Tretjo hipotezo prav tako potrjujemo, saj sta velikost slike in dolžina skritega sporočila neposredno vplivali na stopnjo neopaznosti – večje slike in krajša sporočila so povzročila manj vidne spremembe, medtem ko so manjše slike in daljša sporočila vplivale na opaznejšo spremembo kakovosti.

S tem sva uspešno odgovorila na začetni problem raziskovalne naloge in dokazala, da je metoda LSB primerna za učinkovito in relativno neopazno skrivanje podatkov v slikovne datoteke.

7 POVZETEK

V raziskovalni nalogi sva obravnavala področje steganografije, ki predstavlja tehniko skrivanja informacij znotraj drugih digitalnih nosilcev, kot so slike, video ali zvok. Osredotočila sva se predvsem na slikovno steganografijo in metodo najmanj pomembnega bita (angl. LSB), ki omogoča skrivanje podatkov z minimalnimi spremembami vizualne kakovosti slike.

V praktičnem delu sva razvila program v programskem jeziku C# v razvojnem okolju Visual Studio. Program omogoča skrivanje in branje besedilnih sporočil v slikovnih datotekah formata PNG in BMP. Sporočilo se pred skrivanjem pretvori v binarno obliko, nato pa se njegovi biti vgradijo v najmanj pomembne bite barvnih kanalov (angl. RGB) posameznih pikslov. Program samodejno določi število uporabljenih LSB bitov glede na velikost slike in dolžino sporočila, s čimer zagotovi čim manjše vizualne spremembe.

Rezultati testiranja so pokazali, da so spremembe pri uporabi enega ali dveh LSB bitov za človeško oko praktično neopazne. Ugotovila sva tudi, da velikost slike in dolžina skritega sporočila vplivata na stopnjo neopaznosti. Razvita rešitev je stabilna, uporabniku prijazna in omogoča zanesljivo skrivanje ter branje sporočil.

S tem sva potrdila, da je metoda LSB primerna za učinkovito in relativno diskretno skrivanje podatkov v slikovne datoteke.

8 ZAHVALA

Zahvaljujeva se najinemu mentorju Islamu Mušiću, ki nama je pomagal pri začetku raziskovalne naloge in nama ob koncu pomagal pravilno urediti dokumentacijo.

Zahvaljujeva se tudi Karmen Hudournik, ki je poskrbela za navodila pri izdelavi raziskovalne naloge ter za redno obveščanje o dogodkih, povezanih z raziskovalnimi nalogami.

Seveda se zahvaljujeva tudi najinima staršema, ki nama omogočata šolanje na Šolskem centru Velenje.

9 VIRI IN LITERATURA

[1] W3schools | C# Intro – What is C#?
https://www.w3schools.com/cs/cs_intro.php (7.1.2026)

[2] Medium | How LSB works
https://medium.com/@renantkn/lsb-steganography-hiding-a-message-in-the-pixels-of-an-image-4722a8567046 (7.1.2026)

[3] Stack Overflow | Convert integer to binary in C#
https://stackoverflow.com/questions/2954962/convert-integer-to-binary-in-c-sharp (19.1.2026)

[4] CARNet | Uvod
https://www.cis.hr/www.edicija/LinkedDocuments/CCERT-PUBDOC-2006-04-154.pdf#:~:text=Steganografija%20je%20znanstvena%20disciplina%20koja,Moderna (22.1.2026)

[5] ResearchGate | An Overview of Digital Audio Steganography
https://www.researchgate.net/publication/338029976_An_Overview_of_Digital_Audio_Steganography (13.2.2026)

[6] MDPI | A Review on Text Steganography Techniques
https://www.mdpi.com/2227-7390/9/21/2829 (15.2.2026)

[7] LevelBlue | Image steganography: Concealing secrets within pixels
https://levelblue.com/blogs/levelblue-blog/image-steganography-concealing-secrets-within-pixels (9.2.2026)

[8] ScienceDirect | Video steganography: A review
https://www.sciencedirect.com/science/article/abs/pii/S0925231218312608 (18.2.2026)

10 PRILOGE

Priloga 1 - povezava do programa:

https://github.com/TomazSpegelj/Steganografija

Priloga 2 – Izjava avtorjev:

IZJAVA

Izjavljamo, da smo pri pripravi raziskovalne naloge upoštevali etična načela in smernice v skladu z veljavnimi pravnimi akti raziskovalnega področja.

Podpisani: Lara Koren (avtorica), Tomaž Špegelj (avtor), Islam Mušić (mentor)

Avtorja:

Mentor: