Šolski center Novo mesto
Srednja elektro šola in tehniška gimnazija
Šegova ulica 112
8000 Novo mesto

SPLETNA STRAN ZA UREJANJE FRAKTALOV

(Maturitetna seminarska naloga)

Predmet: Računalništvo
Avtor: Jure Vajs, T4B
Mentor: Gregor Mede, univ. dipl. inž. rač. in inf.

Novo mesto, april 2022

POVZETEK IN KLJUČNE BESEDE

V seminarski nalogi sem opisal izdelavo spletne strani za urejanje fraktalov, ki sem jo izdelal ko zaključni izdelek za maturo, teorijo za fraktali, ki sem jih izdelal, in pa postopek objave spletne strani. To temo sem si izbral zaradi mojega zanimanja za fraktale, in želji po izdelavi le-teh. V prvem delu seminarske sem na kratko opisal vse uporabljene programske jezike, njihov pomen v seminarski nalogi, razložil in opisal matematično teorijo za vsakim izmed fraktalov, in na kratko opisal ponudnike, preko katerih mi je bilo omogočeno objaviti spletno stran . V drugem delu seminarske naloge pa sem opisal samo izdelavo spletne strani, ki sem jo razdelil na backend spletne strani, frontend spletne strani, ter postopek objave spletne strani na internet. Za lažjo predstavo izdelka sem v seminarsko nalogo veliko slik spletne strani, in pa same programske kode, ki spletno stran sestavlja.

Ključne besede:

- Fraktali
- Spletna stran
- P5.js
- JavaScript
- HTML

KAZALA

KAZALO VSEBINE

KAZALO SLIK

KAZALO TABEL

1 UVOD

V modernem času so spletne strani, najlažji in najbolj učinkovit način za prikaz podatkov ali vsebine, ki je dostopen vsem uporabnikom interneta. Spletno stran lahko ustvari in vzdržuje posameznik, skupina, podjetje ali organizacija za različne namene.

V tej seminarski nalogi bom izdelal, in nato opisal postopek izdelave spletne strani. Cilj naloge je izdelati čim bolj uporabnikom prijazno spletno stran, in se preizkusiti v znanju algoritmov in matematike pri izdelavi različnih fraktalov. Pri izdelavi strani se bo najverjetneje pojavilo veliko problemov, s katerimi se bom soočil z svojim znanjem, in pa različnimi viri iz interneta in knjig.

V prvem delu bom na kratko opisal uporabljene programske jezike, teorijo za izdelanimi fraktali, in na kratko opisal uporabljene ponudnike za, ki omogočajo objavo spletnih strani, v drugem delu pa se bom poglobil v samo izdelavo spletne strani, ki jo bom podprl z izseki kode, ki sestavljajo spletno stran, in pa v postopek objave spletne strani.

2 TEORETIČNI DEL

Teoretični del seminarske naloge sestavlja izbira primernih programskih jezikov za izdelavo izdelka in spoznavanje z njimi, ter raziskovanje, spoznavanje in razumevanje fraktalov, ki bodo izdelani. V tem delu bom zato na kratko opisal izbrane programske jezike, teorijo za fraktali, in ponudnike, s pomočjo katerih sem objavil spletno stran na internet.

2.1 HTML IN CSS

HTML in CSS sta jezika, ki skrbita za izgled in postavitev spletne strani. HTML ali Hyper Text Markup Language je označevalni jezik pri katerem se preko značk in atributov, izdelujejo spletne strani. Predstavlja osnovo spletnega dokumenta. Poleg prikaza dokumenta v spletnem brskalniku se z njim hkrati določi tudi zgradba in semantični pomen delov dokumenta (1). CCS ali Cascading Style Sheets, pa je slogovni jezik, ki skrbi za reprezentacijo spletnih strani. Z njim lahko določamo barve, velikosti, odmike, pozicije, in vrsto drugih atributov. CSS nam omogoča zmanjšanje ponavljanja kode, saj omogočimo množici strani uporabo istih podlog, kar lahko bistveno zmanjša njihovo velikost (2).

2.2 JAVASCRIPT

JavaScript je objektni programski jezik, ki je bil razvit z namenom, da bi spletnim razvijalcem pomagal pri ustvarjanju spletnih strani. JavaScript lahko sodeluje s HTML kodo in s tem poživi stran z dinamičnim izvajanjem. Podpirajo ga vsi novejši spletni brskalniki. Sintaksa jezika ohlapno sledi programskemu jeziku C.

JavaScript se veliko uporablja za ustvarjanje dinamičnih spletnih strani. Program se vgradi ali pa vključi v HTML, da opravlja naloge, ki niso mogoče samo s statično stranjo. Na primer odpiranje novih oken, preverjanje pravilnost vnesenih podatkov, enostavni izračuni ipd. Na žalost različni spletni brskalniki izpostavijo različne objekte za uporabo. Za podporo vseh brskalnikov je zato treba napisati več različic funkcij (3).

Programi imajo svoje objektne modele, ki zagotavljajo dostop do gostiteljevega okolja, samo jedro jezika JavaScript pa je v vseh programih večinoma enako (3).

2.3 P5.JS

p5.js je knjižnica JavaScript za kreativno kodiranje, s poudarkom na tem, da je koda dostopna in vključujoča za umetnike, oblikovalce, pedagog, začetnike in vse druge. Je brezplačna in odprtokodna knjižnica, kar pomeni, da je dostopna vsem (4). Temelji na okolju za ustvarjalno programiranje Processing, njegov namen pa je narediti programski jezik bolj prijazen uporabniku s pomočjo vizualizacije. P5.js programi se na splošno imenujejo skice (angl. sketches), saj programerjem omogočajo , da hitro ustvarijo prototipe programske opreme, ali da preizkusijo novo idejo. Vendar pa uporabnik ni omejen samo na risanje na platno (angl. canvas), saj lahko kot svojo skico obravnava celotno spletno stran s pomočjo HTML5 objektov za besedilo, vnos, zvok in spletno kamero.

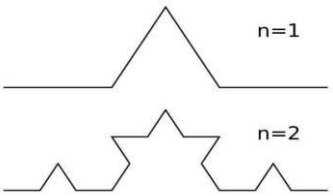
2.4 TEORIJA ZA FRAKTALI

Izraz "fraktal" je skoval matematik Benoit Mandelbrot leta 1975. Mandelbrot ga je utemeljil na latinskem *fractus*, kar pomeni "zlomljen" (5). Fraktalni del je neskončni vzorec. Fraktali so neskončno kompleksni vzorci, ki so sami sebi podobni po različnih lestvicah. Ustvarjajo se tako, da ponavljajo preprost proces znova in znova v zanki povratnih informacij, ki potekajo. Fraktali so slike dinamičnih sistemov – slike Kaosa (6). Geometrično obstajajo med našimi znanimi dimenzijami. Fraktalni vzorci so izredno znani, saj je narava polna fraktalom. Na primer: drevesa, reke, obale, gore, oblaki, školjke, orkani itd. Abstraktne fraktale – kot je Mandelbrotov niz– lahko ustvari računalnik, ki izračuna preprosto enačbo znova in znova (6). Nekateri fraktali obstajajo le iz umetniških razlogov, drugi pa so zelo uporabni. Fraktali so zelo učinkovite oblike za radijske antene in se uporabljajo v računalniških čipih za učinkovito povezavo vseh komponent.

2.4.1 FRAKTALNO DREVO

Fraktalno drevo je eden izmed najpreprostejših primerov fraktala. Definirano je rekurzivno s simetričnim razvejanjem, kar pomeni, da je opredeljeno glede na sebe (prejšnjo generacijo). Vsaka generacija ima dolžino, ki je nek procent dolžine njene predhodne generacije (dolžina nesme biti enaka ali daljša kot njena predhodna dolžina). Edina izjema je deblo drevesa, ki ima podano dolžino.

2.4.2 KOCHOVA KRIVULJA



Slika 1: Razvoj Kochove krivulje (18)

Kochova krivulja je matematična krivulja in ena od najstarejših fraktalnih krivulj, ki so bile opisane. Prvič se je pojavila v papirju iz leta 1904 z naslovom "On a continuous curve without tangents, constructible from elementary geometry" švedskega matematika Helga von Koch-a (7).Sestavimo jo lahko s tremi koraki. Najprej razdelimo odsek črte na tri enake dele, nato narišemo enakostranični trikotnik, ki ima za osnovnico srednji del na razdeljenem segmentu, in je obrnjen navzven, na koncu pa odstranimo del, ki je osnovnica trikotnika, kot je vidno na sliki 1.

2.4.3 MANDELBROTOV NIZ

$$z_{n+1} = z_n^2 + c \qquad \begin{aligned} z_0 &= 0 \\ {z_0}^2 + c &= z_1 \\ {z_1}^2 + c &= z_2 \\ {z_2}^2 + c &= z_3 \end{aligned}$$

Slika 2: Funkcija Mandelbrotovega seta in vrednost z-ja skozi iteracije

Mandelbrotov niz je množica kompleksnih števil c, za katera funkcija, ki je vidna na sliki 2 ne zdrsne v neskončnost. Števila ki ne zdrsnejo v neskončnost predstavljajo orednji del niza. Števila ki zdrsnejo, pa lahko obarvamo po različnih sistemih barvanja, ki jim pripišejo barvno vrednost glede na število iteracije, pri kateri so zdrsnila v neskončnost. Mandelbrotov niz izvira iz področja kompleksne dinamike, ki sta ga na začetku 20. Stoletja raziskovala francoska matematika Pierre Fatou in Gaston Julia. Ta fraktal sta leta 1978 prvič opredelila in narisala Robert W. Brooks in Peter Matelski kot del študije Kleinianovih skupin (8). Matematično preučevanje Mandelbrotovega niza se je v resnici začelo z delom matematikov Adriena Douadyja in Johna H. Hubbarda, ki sta postavila številne njegova temeljne lastnosti in množico poimenovala v čast Mandelbrota za njegovo vplivno delo na področju fraktalne geometrije (9) (10).

2.5 HOSTGATOR

HostGator je houstonski ponudnik skupnih, prodajnih, virtualnih zasebnih strežnikov in namenskega spletnega gostovanja z dodatno prisotnostjo v Austinu v Teksasu (11). HostGator je bil oktobra 2002 ustanovljen s strani Brenta Oxleya, ki je bil takrat študent na univerzi Florida Atlantic University. Leta 2006 se je podjetje preselilo iz prvotne pisarne v Boca Ratonu na Floridi v novo stavbo na 20.000 kvadratnih metrih v Houstonu v Teksasu. Junija 2006 je podjetje odprlo svojo prvo mednarodno pisarno v Kanadi (11).

2.6 CPANEL

cPanel je programska oprema nadzorne plošče za spletno gostovanje, ki jo je razvil cPanel, LLC. Ponuja grafični vmesnik (GUI) in orodja za avtomatizacijo, zasnovana za poenostavitev postopka gostovanja spletnega mesta lastniku spletne strani ali »končnemu uporabniku«. Omogoča administracijo prek standardnega spletnega brskalnika z uporabo tritierne strukture. Medtem ko je cPanel omejen na upravljanje enega gostiteljski račun, cPanel & WHM omogoča upravljanje celotnega strežnika (12).

3 PRAKTIČNI DEL

Za izdelavo fraktalov sem se odločal med Programskim jezikom Processing in pa p5.js, ki je knjižnica vgrajena v programski jezik JavaScript, in je bila narejena po sledeh Processinga. Na koncu sem se odločil za p5.js, saj mi je omogočal programiranje v okolju Vscode, kar je pomenilo da sem imel vse datoteke v isti mapi, in pa lažjo vključitev p5.js skic v spletno stran. Po tem, ko pa sem se odločil za programske jezike, ki so bili po mojem mnenju najbolj primerni za izdelavo izdelka in po tem ko sem se spoznal z fraktali, sem začel z izdelavo splene strani.

3.1 SINTAKSA

3.1.1 JAVASCRIPT

Ker sem vse skice pisal po p5.js strukturi programa, sem rabil le tiste primarne JavaScript ukaze, ki so mi omogočali branje vrednosti iz vnosnih mest in drsnikov, in pa ukaz ki omogoča vpise določenih vrednosti v konzolo. Tabela 1 prikazuje primarna JavaScript ukaza, ki sem ju uporabil.

Tabela 1: JavaScript ukaza

document.getElementBy().value	Prebere vrednost gumba po njegovem ID-ju, in ga pripiše določeni spremenljivki.
console.log()	Vpiše vrednost spremenljivke med oklepaji v terminal.

3.1.2 P5.JS

Kljub temu da je p5 knjižnica JavaScripta, ima zaradi svoje grafične naklonjenosti nekoliko drugačno sestavo programa. Vsak program je sestavljen iz funkcije setup(), ki se izvede prva ob zagonu programa, a le enkrat, in funkcije draw(), ki se izvede za funkcijo setup, in se ponavlja neprestano. Tabela 2 prikazuje glavne funkcije p5,js knjižnice.

Tabela 2: Glavni funkciji p5.js knjižnice

function setup()	Pokliče se le enkrat, in sicer ob zagonu programa. Uporablja se za deifiniranje začetnih lastnosti skice (13).
function draw()	Klicana neposredno po setup() funkciji. Deluje neprekinjeno do ustavitve programa ali do klica noLoop() funkcije.

Ostale funkije ki niso primarne, a sem jih uorabil pri vseh skicah:

Tabela 3: Ostale uporabljene funkcije p5.js knjižnice

windowResized()	Se izvede takrat, ko se spremeni višina ali širina spletne strani.
resizeCanvas()	Omogoča spreminjanje velikosti platna. Večinoma uporabljena znotraj windowResized() funkcije.
CreateCanvas()	Ustvari platno z podano širino in višino.
canvas.parent()	Znotraj oklepajev se vpiše id HTML div-a, preko katerega lahko lažje prilagajamo pozicijo in lastnosti skice na strani.
angleMode()	Nastavi mersko enoto za kote. Možnosti izbire med radiani in stopinjami (14).
translate()	Prestavi izhodiščno točko risanja levega zgornjega kota na željeno točko.
rotate()	Zavrti črto ali lik za določeno vrednost.
push()	Shrani trenutne nastavitve sloga, risanja in transformacije (15).
pop()	Shranjene nastavitve iz push() funkcije obnovi. Uporablja se vedno v kombinaciji z funkcijo push() (16).

3.2 PROGRAMIRANJE FRAKTALOV

Vseh fraktalov sem se načeloma lotil na isti način. Najprej sem si prebral članke in teorijo o določenem fraktalu, nato pa sem po svojih najboljših močeh poskusil ta fraktal sprogramirati sam. Ko sem naletel na kakšen problem ali zanko, ki je nisem znal sam razvozlati, sem se obrnil na pomoč virov iz interneta, ki so mi močno pomagali.

3.2.1 DREVO

3.2.1.1 Z UREJANJEM

Ker je bil ta fraktal najbolj preprost izmed treh, sem se ga lotil prvega. Po tem ko, je v funkciji setup podana širina, višina in pa odzadje platna, program bere uporabnikov vnos za kot alfa in kot beta drevesa kot vidimo na sliki 3. Kot alfa predstavlja odmik desne veje glede na osnovnico, kot beta pa odmik leve veje glede na osnovnico.

```
alpha = document.getElementById("kotalfa").value;
console.log(alpha);
beta = document.getElementById("kotbeta").value;
console.log(beta);
```

Slika 3: Branje vrednosti za kote

Za tem sem z funkcijo translate poiskal sredino spodnje stranice platna, in podal barvo in debelino risanja, kot je vidno na sliki 4.

```
translate(windowWidth/6,windowWidth/4);
stroke (255);
strokeWeight(4);
```

Slika 4: Translacija in podajanje barve in debeline risanja

Nato sem klical funkcijo generiranje(), ki jo vidimo na sliki 5, pri kateri se je začelo risanje drevesa. Glede na podano dolžino pri klicanju, je funkcija narisala iz izhodišča (ki je bilo v prejšnem koraku spremenjeno) črto, in s funkcijo translate() prestavila izhodišče na zgornji konec narisane črte. Nato gre funkcija v if zanko, v kateri se razdeli s funkcijama push() in pop() na dve veji. Pri vsaki izmed vej se najprej pot zavrti za določeni kot (pri vsaki je ta kot drugačen), nato pa se funkcija generiranje kliče sama v sebi z dolžino, ki je nek delež stare dolžine, in ustvari neskončno zanko, ki pa se zaradi if stavka izvaja le dokler je dolžina nad vrednostjo 9.

```
function generiranje(lenght){
    line(0,0,0,-lenght);
    translate(0,-lenght);
    if(lenght>9)
    {
        push();
        rotate(alpha);
        generiranje(lenght * 0.66);
        pop();
        push();
        rotate(beta);
        generiranje(lenght * 0.66);
        pop();
    }
}
```

Slika 5: Funkcija generiranje

3.2.1.2 NA ZAČETNI STRANI

Zaradi majhne računske zahtevnosti tega fraktala, je bil najbolj primeren za prikaz na začetni strani, ki bi vključeval zelo hitro in zanimivo interaktivnost. Kot alfa in beta je program zato namesto iz vnašanja števil bral iz koordinatske pozicije kazalca računalniške miške na strani, kot je vidno na sliki 6.

```
function generiranje(lenght){

    line(0,0,0,-lenght);
    translate(0,-lenght);
    if(lenght>9)
    {
        push();
        rotate((mouseX/10)/(TWO_PI*8));
        generiranje(lenght * 0.66);
        pop();
        push();
        rotate(-(mouseX/10*0.3)/(TWO_PI*8));
        generiranje(lenght * 0.66);
        pop();
    }

}
```

Slika 6: Funkcija generiranje (začetna stran)

3.2.2 KOCHOVA RAVNINA

Kot drugega fraktala sem se lotil Kochove ravnine. Po deklaraciji platna v funkciji setup(), sem nato v funkciji draw() deklariral dva vektorja, in jima določil izhodiščni koordinati, kot je vidno na sliki 7.

```
let a = createVector(0,height/1.5);
let b = createVector(width,height/1.5);
```

Slika 7: Deklaracija vektorjev

Za deklaracijo vektorjev program bere uporabnikov vnos kota obračanja in pa števila generacij ravnine, kot prikazuje slika 8.

```
koch_kot = document.getElementById("kotobracanja").value;
console.log(koch_kot);
stgeneracij = document.getElementById("stgeneracij").value;
console.log(stgeneracij);
```

Slika 8: Branje vrednosti kota in števila generacij

Ker se računska zahtevnost tega fraktala z vsako generacijo večja za 4-krat, se lahko to hitro pozna pri hitrosti izrisa ravnine. Zaradi tega razloga sem za branjem vrednosti vstavil if stavek, ki je prikazan na sliki 9, ki v primeru da je vpisano število generacij večje od 5 pokaže okensko opozorilo na strani, in vrednost števila generacij ponastavi na vrednost, ki je bila vpisana pred to.

```
if(stgeneracij > 5){
    stgeneracij = stgeneracijStaro;
    document.getElementById("stgeneracij").value = stgeneracijStaro;
    window.alert('Izbrali ste število izven mej!')
}
```

Slika 9: Opozorilo ob preveliki vrednosti števila generacij

Po deklaraciji vektorjev sem iz njiju deklariral nov objekt iz razreda del, kot lahko vidimo na sliki 10, v katerem sem računal koordinate posameznih delov ravnine in pa izrisoval te dele, nato pa ta objekt shranil v tabelo delov z funkcijo .push().

```
let v = new del(a,b);
ravnina.push(v);
```

Slika 10: Deklaracija objekta iz razreda "del"

Nato gre program v for zanko, prikazano na sliki 11, ki se izvede le tolikokrat, kot je določeno število generacij v programu. V zanki se nato najprej deklarira tabela “generacija”, ki bo prenašala v vse koordinate v tabelo “ravnina”, in je v zanki deklarirana, ker je potrebno da je za izris vsake generacije prazna, saj se le nekatere koordinate različnih generacij ujemajo. Zanka gre nato še v eno zanko, ki pa se izvaja toliko, kolikor je dolžina tabele “ravnina” (vsak indeks v tabeli “deli” predstavlja eno črto). V tej zanki se nato najprej z metodo “računanje” izračuna razdelitev črte in se shrani v tabelo “odsek”, nato pa se te koordinate z funkcijo “dodaj” le dodajo v tabelo “generacija”, ki ima po izvedbi notranje for zanke shranjene vse koordinate za trenutno število generacij, in se nato v celoti prenese v tabelo deli, ki bo po končani zunanji zanki izrisana.

```
for(let x = 0; x<stgeneracij; x++){
    let generacija = [];
    for(let y = 0; y<ravnina.length; y++){
    odsek = ravnina[y].racunanje();
        dodaj(odsek,generacija);
    }
    ravnina = generacija;
}
```

Slika 11: Priprava krivulje za izris

Že prej omenjena metoda “racunanje”, ki je vidna na sliki 12, torej za vsak indeks v tabeli “ravnina” izračuna koordinate ali vektorje za naslednjo generacijo. Vsak izmed štirih delov, ki so izdelani v tej metodi, se shrani v tabelo deli, kot nov objekt v tabelo “ravnina”. Funkcija najprej izračuna dolžino začetne črte tako, da z funkcijo p5.Vector.sub() odšteje točko a od točke b, nato pa jo za nadaljnje računenje deli z tri, z funkcijo .div(). Nato z metodo add sešteje začetno točko a in dolžino, in ustvari vektor “prvi_del”, ki bo predstavljal konec prve črte. V tabelo koordinate nato pod ničti indeks zapiše nov objekt (črto/vektor), z začetkom v točki a in koncem v točki “prvi_del”. Podobno naredi tudi za četrti del, za drugi intretji del, ki pa sta rotirana, pa le ustvari vektor “drugi_del”, ki s pomočjo vnesene vrednosti za rotacijo kota uporabnika zarotira smer vektorja. Pred vrednostjo kota je minus, ker je mreža v programu obrnjena na glavo. Metoda po izračunu vseh koordinat le še vrne tabelo “koordinate”.

```
racunanje(){

    let koordinate = [];
    angleMode(DEGREES);

    let dolzina = p5.Vector.sub(this.tocka_b,this.tocka_a);
    dolzina.div(3);

    let prvi_del = p5.Vector.add(this.tocka_a,dolzina);
    koordinate [0] = new del(this.tocka_a, prvi_del);

    let tretji_del = p5.Vector.sub(this.tocka_b,dolzina);
    koordinate [3] = new del(tretji_del, this.tocka_b);

    dolzina.rotate(-koch_kot);
    let drugi_del = p5.Vector.add(prvi_del, dolzina);

    koordinate [1] = new del(prvi_del, drugi_del);
    koordinate [2] = new del(drugi_del, tretji_del);


    return koordinate;
}
```

Slika 12: Računanje koordinat krivulje

Po tem ko imamo izračunane vse vektorje, nam preostane le še izris, ki ga dosežemo z zanko, ki gre skozi vse elemente tabele “deli”, in pa funkcije “risanje”, prikazane na sliki 13, ki bo vsak vektor posebej izrisala.

```
risanje(){
    stroke(255);
    line(this.tocka_a.x, this.tocka_a.y, this.tocka_b.x, this.tocka_b.y);
}
```

Slika 13: Risanje krivulje

Ker pa hočemo, da se vsak izris avtomatko osveži, ko spremenimo katerokoli izmed vrednosti, je celotna koda za računanje zapisana v funkciji draw(), kar pomeni da se neprestano ponavlja. Zato moramo na koncu funkcije draw() le še izprazniti tabelo “ravnina”, da nebi pri naslednjem izrisu narisala še prejšnjo variacijo ranvnine.

3.2.3 MANDELBROTOV NIZ

Kot zadnjega, in najbolj zahtevnega fraktala za razumevanje sem se lotil Mandelbrotega niza, ki pa sem ga izdelal na nekoliko drugačen način kot ostala dva. Ker je fraktal opisan z rekurzivno funkcijo, je bilo najbolj primerno delo z piksli, kjer bo vsak piksel na skici predstavljal eno točko v koordinatnem sistemu.

Program v funkciji setup po deklaraciji platna in podaji velikosti le-tega, prvič pokliče funkcijo generiranje(), ki nariše prvotni izgled fraktala, in se izvede le enkrat. Nato pa v funkciji draw(), ki je prikazana na sliki 14bere uporabnikov vnos vseh parametrov, ki jih je možno urejati, in kliče funkcijo generiranje(), le v primeru, da je vsaj ena izmed štirih vrednosti spremenjena, saj bi bilo risanje nove slike ob nespremenjenih parametrih odveč.. Na koncu le še vse vrednosti parametrov shrani pod stare vrednosti, da jih naslednjo ponovitev lahko primerja z novimi vrednostmi, in preverja ali so spremenjene.

```
function draw(){
    svetilnost = document.getElementById("svetilnost").value;
    red = document.getElementById("R").value;
    green = document.getElementById("G").value;
    blue = document.getElementById("B").value;

    if(redStaro != red || svetilnostStaro != svetilnost || greenStaro != green || blueStaro!= blue){
        generiranje();
    }
    svetilnostStaro = svetilnost;
    redStaro = red;
    greenStaro = green;
    blueStaro = blue;
}
```

Slika 14: Branje vrednosti uporabnika, klicanje funkcije generiranje()

Funkcija generiranje() torej kot že prej omenjeno opravlja računanje točk, barvanje in risanje fraktala. Ker si bomo skico predstavljali kot nekakšen koordinatni sistem, v katerem bo izrisan “graf” fraktala, gre program nato v dve for zanki. Prva for zanka bo šla čez vse x koordinate osi, druga pa čez vse y koordinate grafa. Za vsako izmed točk sestavljenih iz teh x in y koordinat bomo v nadaljenavnju z pomočjo funkcije preverjali, ali spada v notranjost fraktala, ali zdrsne v neskončnost, in predstavlja zunanjost fraktala. V zanki bomo najprej priredili vrednost x iz med 0 in širino platna na med -2 in 1.5 in jo shranili pod a, vrednost y pa bomo priredili iz med 0 in višino platna na med -1.5 in 1.5. Vrednosti niso iste, saj Mandelbrotov niz v kvadratu zgleda sploščeno. Nato bomo deklarirali spremenljivki “a_od_c” in “b_od_c”, ter a shranili pod prvo, b pa pod drugo. To naredimo, saj bomo pri vsaki iteraciji z-ja sešteli kvadrat z-ja prejšne iteracije, in pa začetnega kompleksnega števila. Torej bo “a_od_c” predstavljal realno komponento začetnega kompleksnega števila, “b_od_c” pa bo predstavljal imaginarno komponento kompleksnega števila. Postopek je prikazan na sliki 15.

```
let a = map(x, 0, width, -2, 1.5);
let b = map(y, 0, height, -1.5, 1.5);

let stevec = 0;
let a_od_c = a;
let b_od_c = b;
```

Slika 15: Prirejanje vrednosti, in deklaracija komponent kompleksnega števila

Ker bomo v nadaljevanju rabili za vsako izmed teh vrednosti a in b preverjati ali zdrsneta v neskončnost, napišemo for zanko prikazano na sliki 16, v kateri računamo tudi kvadrat z-ja prejšnje generacije. Že iz prvih nekaj generacij iteracij kot je vidno na sliki 17 ugotovimo, da je potrebno izračunati le kvadrat trenutnega kompleksnega števila, in mu prišteti a in b vrednosti začetnega kompleksnega števila. Po kratkem računu ugotovimo, da je kvadrat kompleksnega števila na sliki 18 isto kot poenostavljen kvadrat kot prikazuje slika 19. Kvadrat kompleksnega števila iz slike 19 pa lahko nato spet razdelimo na realno komponento, kot je razvidno na sliki 20, in pa imaginarno komponento, kot je vidno na sliki 21, kar pomeni da lahko realno in imaginarno komponento računamo posebej, in ju združimo za tem.

```
for(stevec; stevec<stevecmax; stevec++){
    let a_na2 = a*a - b*b;
    let b_na2 = 2*a*b;

    a = a_na2 + a_od_c;
    b = b_na2 + b_od_c;
```

Slika 16: Računanje z-ja trenutne iteracije

$$(z_0 = 0;\ z_1 = z_0^2 + c = c;\ z_2 = c^2 + c;\ z_3 = (c^2 + c)^2 + c)$$

Slika 17: Prvih nekaj iteracij Mandelbrotovega niza

$$c^2 = (a + bi)(a + bi)$$

Slika 18: Kvadrat kompleksnega števila

$$c^2 = a^2 - b^2 + 2abi$$

Slika 19: Poenostavljen kvadrat kompleksnega števila

$$(a^2 - b^2)$$

Slika 20: Realna komponenta kvadrata kompleksnega števila

$$(2abi)$$

Slika 21: Imaginarna komponenta kvadrata kompleksnega števila

Nato pridemo v fazo preverjanja, ali je število zdrsnilo v “neskončnost” ali ne, kot je vidno na sliki 17. Neskončnost je bila v mojem primeru definirana z 2, saj je pri takem številu iteracij program lahko zelo počasen. Če je torej število število večje od 2, bomo sklepali, da je zdrsnilo v neskončnost in izstopili iz zanke, števec pa se bo shranil, in nam pozneje pomagal pri določanju barve te točke.

```
if(sqrt(a*a + b*b)>2){
    break;
}
```

Slika 22: Preverjanje zdrsa v neskončnost

Ko pridemo iz zanke, ki preverja števila, vstopimo v sistem barvanja, prikazanega na sliki 18, ki sem ga zasnoval sam. Vse kar naredi je da preveri velikost števca pri posameznih točkah. Če je imelo število števec 0, je to pomenilo da je izpadlo že v prvi iteraciji, in ga obarvamo temno, a ne črno, če pa je število prišlo do zadnje iteracije, točko obarvamo popolnoma črno. Vsa števila umes pa nekoliko priredimo vrednosti števca, a zaradi %255 ki je uporabljen na koncu računanja stevlosti, so lahko pri nekaterih slikah sosednje barve abstraktne, saj je brez pomena uporabljati vrednosti večje od 255, ko RGB sistem barvanja sprejme le vrednosti med 0 in 255.

```
let svetlost = (stevec*svetilnost)%255;

if(stevec === 0){
    svetlost = svetilnost-10;
}

if(stevec === stevecmax){
    svetlost = 0;
}

rd= (svetlost*red);
gr = (svetlost*green);
bl = (svetlost*blue);
```

Slika 23: Sistem barvanja

Na koncu le še uporabimo tabelo pixels, ki je vgrajena v knjižnico. Tabela gre čez vse piksle na platnu, in ga obarva glede na RGB vrednost. Kjer so vrednosti vseh R, G in B komponent 0, bo točko obarvalo črno, kjer so vse 255, bo točko obarvalo belo, ostale pa nekje umes. Del programa je viden na sliki 19.

```
var index = (x + y * (int)(width)) * 4;
pixels[index + 0] = rd;
pixels[index + 1] = gr;
pixels[index + 2] = bl;
pixels[index + 3] = 255;
```

Slika 24: Barvanje pisklov

3.3 OBLIKOVANJE SPLETNE STRANI

Za oblikovanje izgleda spletne strani sem uporabljal programska jezika HTML in CSS, ki sta načeloma standarda za opravljanje takih nalog. Ker sem vedel da bodo fraktali na spletni strani izgledali precej barvni in ”glasni”, sem probal ostale dele spletne strani izdelati čim bol umirjene in monotone. Pri vseh straneh razen pri glavni sem p5.js platna postavil na levo stran stran, pri glavni pa sem bolj razvlečeno platno postavil na sredino spletne strani. Ker so platna sama po sebi zelo neprilagodljiva za postavitev na strani, sem vsakemu platnu z metodo .parent() v p5.js datoteki določil, HTML kot je prikazano na sliki 20, preko katerega sem jih v HTML datoteki klical kot je vidno na sliki 21, in jim v CSS datoteki dajal ukaze za izgled in postavitev, kot je prikazano na sliki 22.

```
canvas.parent('mandelcustomizable');
```

Slika 25: Pripisovanje platna HTML div-u

```
<div id="mandelcustomizable"></div>
```

Slika 26: Klicanje platna preko HTML div-a

```
canvas {
  border: 6px solid #ffffff;
}
#mandelcustomizable{
  display: flex;
  justify-content: left;
  align-items: left;
  padding-left: 3%;
}
```

Slika 27: CSS nastavitve za izgled in postavitev platna

Pri strani za fraktalno drevo in kochovo ravnino, sem za uporabnikovo možnost za urejanje le-teh uporabil vnosna mesta kamor lahko uporabnik z tipkovnico vpisuje različne vrednosti. Vsa vnosna mesta skupaj pa sem zapakiral v tabelo kot je vidno na slikah 23 in 24, saj mi je to omogočalo najbolj enostavno oblikovanje in razporejanje le-teh.

```
<table>
  <tr>
    <td align="left"><label for="kotalfa">Kot alfa (stopinje): </label></td>
    <td align="left"><input type="number" id="kotalfa" value="28" placeholder="alfa (stopinje)"></td>
  </tr>
  <tr>
    <td align="left"><label for="kotbeta">Kot beta (stopinje): </label></td>
    <td align="left"><input type="number" id="kotbeta" value="-28" placeholder="beta (stopinje)"></td>
  </tr>
</table>
```

Slika 28: HTML koda za vnosna mesta

Slika 29: Izgled vnosnih mest na spletni strani

Pri strani za Mandelbrotov niz pa se mi je zdelo, da bodo najboljši prikaz vseh možnih barvnih kombinacij niza omogočali drsniki. Isto kot pri vnosnih mestih pa sem jih za enostavnejše oblikovanje in razporejanje vse zapakiral v tabelo kot je prikazano na slikah 25 in 26.

```
<table>
  <tr>
    <td align="left"><label for="svetilnost">Svetlost: </label></td>
    <td align="left"><input type="range" id="svetilnost" value="40"  min="0" max="254" placeholder="svetlost"></td>
  </tr>
  <tr>
    <td align="left"><label for="R">Vrednost rdeče: </label></td>
    <td align="left"><input type="range" id="R" value="0" min="0" max="25" placeholder="vrednost rdeče"></td>
  </tr>
  <tr>
    <td align="left"><label for="G">Vrednost zelene: </label></td>
    <td align="left"><input type="range" id="G" value="2" min="0" max="25" placeholder="vrednost zelene"></td>
  </tr>
  <tr>
    <td align="left"><label for="B">Vrednost modre: </label></td>
    <td align="left"><input type="range" id="B" value="15" min="0" max="25" placeholder="vrednost modre"></td>
  </tr>
</table>
```

Slika 30: HTML koda za drsnike

Slika 31: Izgled drsnikov na spletni strani

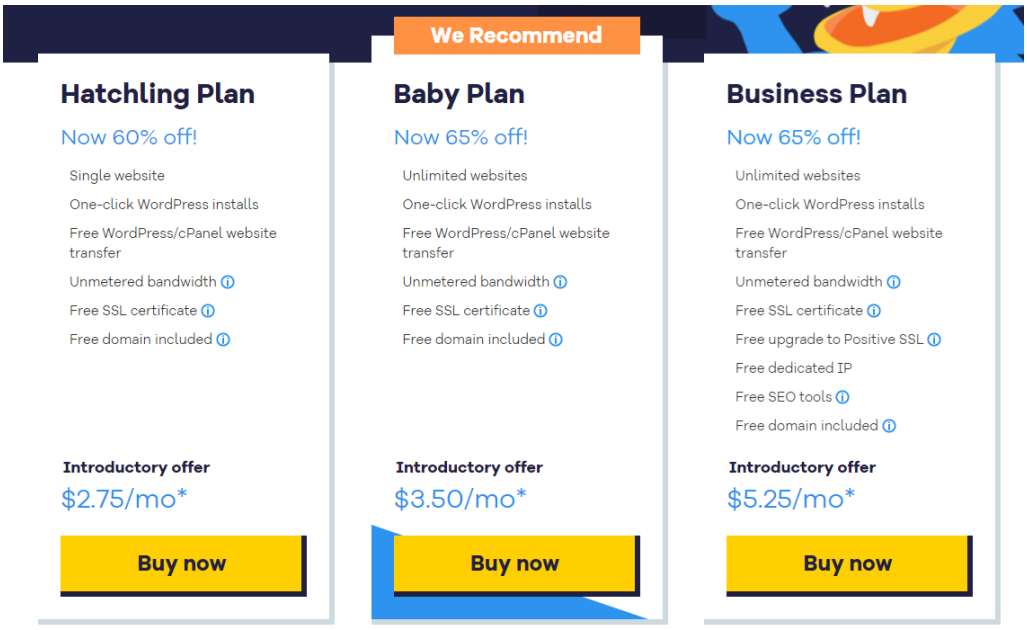
Za popoln izgled spletne strani sem za vsemi platni, drsniki in vnosnih mestih dodal le še navigacijsko vrstico, naslov, in pa vsebino ki je uporabniku fraktal na kratko opisala. Celotno spletno stran lahko najdete pod prilogami, Objavljen pa je tudi link do Git Hub profila, kjer se nahaja celotna koda spletne strani.

3.4 OBJAVA SPLETNE STRAN

Po nekaj raziskovanja po internetu, sem zasledil pa ponudnih spletnega gostovanja HostGator ponuja v vseh njihovih možnih načrtih zastonj domeno in pa SSL certifikat, ki naj bi domeno avtomatsko preusmerjal iz http naslova na https naslov. Po cenovnem primerjano z ostalimi ponudniki sem se zato nato odločil za HostGator, saj mi je bila

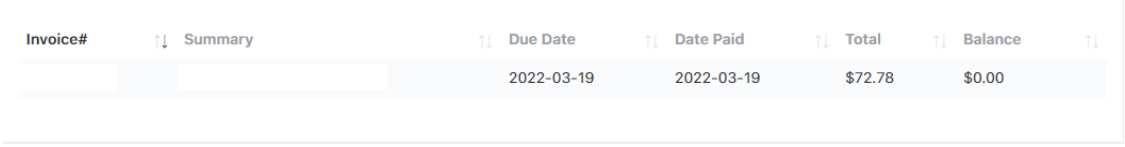
za objavo statične spletne strani, ki ne vsebuje nikakršnih zasebnih podatkov nekoliko bolj pomembna nižja cena kot varnost.

Po izdelavi računa na spletni strani HostGatorja sem se nato med tremi ponudbami ki jih ponujajo, kot je prikazano na sliki 27, odločil za “Hatchling plan”, saj je imel po mojem mnenju vse kar sem potreboval za objavo moje spletne strani.



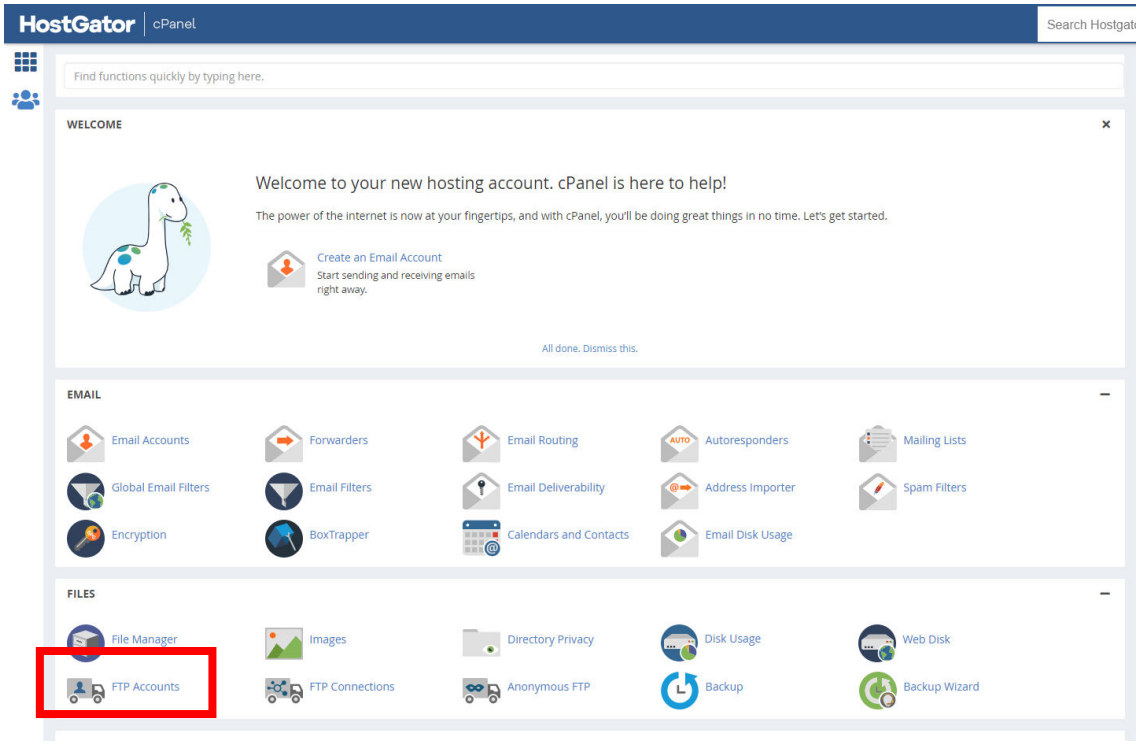
Slika 32: Ponudba načrtov HostGatorja

Po vseh stroških, ki so prišli z objavo spletne strani, me je vse skupaj prišlo 72,78 dolarjev, kot je prikazano na sliki 28, kar je po mojem mnenju za objavo statične spletne strani, saj velja to plačilo le za 12 mesecev ali 1 leto.



Slika 33: Celotni znesek plačila

Ko sem opravil plačilo, sem nato dobil dostop do Cpanel-a, kjer se zares začne objava spletne strani. Po odprtju Cpanel-a se odpre stran kjer lahko izbiraš med mnogimi možnostmi za objavo spletne strani, kot prikazuje 29.



Slika 34: Cpanel

Ker je moja spletna stran statična, je bila moja naloga le, da preko upravitelja datotek (slika-) v datoteko public_html objavim moje datoteke za spletno stran, kot lahko vidimo na sliki 30.

.well-known	4 KB	Mar 19, 2022, 8:42 PM	httpd/unix-directory	0755
cgi-bin	4 KB	Mar 20, 2022, 9:01 PM	httpd/unix-directory	0755
libraries	4 KB	Nov 9, 2021, 3:47 AM	httpd/unix-directory	0755
.htaccess	0 bytes	Apr 10, 2022, 11:25 PM	text/x-generic	0644
drevo.css	3.43 KB	Mar 19, 2022, 6:38 PM	text/css	0644
drevo.html	2.33 KB	Mar 20, 2022, 10:04 PM	text/html	0644
drevo.js	704 bytes	Mar 5, 2022, 1:47 AM	text/x-generic	0644
drevocustomizable.js	1.33 KB	Mar 31, 2022, 6:40 AM	text/x-generic	0644
favicon.png	6.43 KB	Mar 20, 2022, 10:03 PM	image/x-generic	0644
Fractals_website-main.zip	247.8 KB	Apr 10, 2022, 4:26 PM	package/x-generic	0644
googlef42828b1f58cfe9f.html	53 bytes	Apr 10, 2022, 4:30 PM	text/html	0644
index.html	1.39 KB	Apr 10, 2022, 4:32 PM	text/html	0644
j.js	51 bytes	Apr 10, 2022, 4:15 AM	text/x-generic	0644
jsconfig.json	155 bytes	Nov 9, 2021, 3:47 AM	text/x-generic	0644
koch.html	2.92 KB	Mar 20, 2022, 10:04 PM	text/html	0644
koch_ravnina.js	1.64 KB	Mar 19, 2022, 8:07 PM	text/x-generic	0644
koch_ravnina_del.js	1.023 bytes	Feb 17, 2022, 4:28 AM	text/x-generic	0644
mandel.html	3.46 KB	Mar 20, 2022, 10:04 PM	text/html	0644
mandelbrot_set.js	2.45 KB	Mar 19, 2022, 8:08 PM	text/x-generic	0644
README.md	222 bytes	Mar 20, 2022, 10:24 PM	text/x-generic	0644
style.css	1.04 KB	Mar 19, 2022, 2:37 AM	text/css	0644

Slika 35: Upravitelj datotek na Cpanel-u

Po tem ko sem objavil vse datoteke, sem nekaj ur probaval vzpostaviti SSL certifikat, ki mi je bil obljubljen ob nakupu načrta, a brez uspeha. Nato sem na internetu naletel na članek, ki mi je pomagal z ustvarjanjem .htaccess datoteke narediti isto stvar, ki jo dela SSL certifikat z nekaj vrsticami kode, in pa skrajšati link, ki se pokaže kot URL spletne strani, kar je razvidno na sliki 31.

```
# Always use https for secure connections
# Replace 'www.example.com' with your domain name
# (as it appears on your SSL certificate)
RewriteEngine On
RewriteRule ^index\.html$ / [R=301,L]
RewriteCond %{SERVER_PORT} 80
RewriteRule ^(.*)$ https://jvfractals.com/$1 [R=301,L]
```

Slika 36: .htaccess file

(17)

4 ZAKLJUČEK

Kljub temu, da sem spletno stran začel izdelovati že v začetku, šolskega leta, je bila pot do končnega izdelka zelo dolga in izčrpajoča, saj je bila edina stvar ki smo jo delali v šoli, ki se je navezovala na mojo spletno stran osnovno oblikovanje strani z uporabo HTML-ja, ki smo se jo učili v tretjem letniku pri predmetu računalniški sistemi in omrežja. Kljub temu, da je bila pot do zaključka dolga, je bila izdelava spletne strani in pa predvsem fraktalov zelo zanimiva, zato sem z zaključenim izdelkom tudi zadovoljen. Velikokrat sem tudi že po izdelavi kakšnega fraktala več ur sedel za računalnikom in se igral in preizkušal različne barvne kombinacije. Vse napisane programe sem potrudil opisati po svojih najboljših močeh, in upam da so vam bile razlage jasne in razumljive. Za samo izdelavo spletne strani me je navdihnila knjiga Benoita Mandelbrota “The fractal geometry of nature”, iz katere pa razen navdiha za barvne kombinacije na žalost nisem odnesel kaj več, saj se spušča veliko globje v matematiko fraktalov, kot sem lahko sam razumel. Za večino virov sem torej uporabil spletne strani, in vire iz interneta, saj so bili najbolj dostopni, in najlažji za razumevanje. Pri izdelavi spletne strani sem se veliko naučil o fraktalih in njihovem obnašanju in pa seveda o izdelavi spletnih strani in objavi, le teh na spletu. Na koncu sem razmišljal o dodajanju baze podatkov za prijavo in shranjevanje izdelanih fraktalov, a se po posvetu z mentorjem za to nisem odločil. V prihodnje bom še veliko eksperimentiral z različnimi načini barvanja, in pa z različnimi fraktali, moj namen pa je v kratkem dodati še del spletne strani, kjer bi uporabnik lahko ustvarjal svoje fraktale.

5 ZAHVALA

Za konec seminarske naloge bi se rad še zahvalil mentorju Gregorju Medetu, univ. dipl. inž. rač. in inf., ki mi je bil pripravljen vedno pomagati in mi dati kakšen nasvet ali me spraviti na pravo pot, ko nisem bil prepričan kaj bi pri seminarski lahko še dodal, kljub temu da večino snovi ki sem jo potreboval nismo delali v šoli, in pa dr. Albertu Zorku univ. dipl. inž. el., ki me je skupaj z profesorjem Gregorjem Medetom v štirih letih šolanja na Tehniški Gimnaziji v Novem mestu naučil vseh osnov programiranja in računalniških osnov na splošno.

6 VIRI IN LITERATURA

1. **Wikipedija.** HTML. *Wikipedija.* [Elektronski] 23. september 2021. https://sl.wikipedia.org/wiki/HTML.

2. —. CSS. *Wikipedija.* [Elektronski] 19. avgust 2020. https://sl.wikipedia.org/wiki/CSS.

3. —. JavaScript. *Wikipedija.* [Elektronski] 19. marec 2021. https://sl.wikipedia.org/wiki/JavaScript.

4. **GeeksforGeeks.** p5.js. *GeeksforGeeks.* [Elektronski] 17. december 2021. https://www.geeksforgeeks.org/p5-js/.

5. **Wikipedia.** Fractal. *Wikipedia.* [Elektronski] 4. april 2022. https://en.wikipedia.org/wiki/Fractal.

6. **Fractal Foundation.** What are Fractals. *Fractal Foundation.* [Elektronski] 2018. https://fractalfoundation.org/resources/what-are-fractals/.

7. **GeeksforGeeks.** Koch Curve or Koch Snowflake. *GeeksforGeeks.* [Elektronski] 3. oktober 2018. https://www.geeksforgeeks.org/koch-curve-koch-snowflake/.

8. **Wikipedia.** Mandelbrot set. *Wikipedia.* [Elektronski] 21. marec 2022. https://en.wikipedia.org/wiki/Mandelbrot_set.

9. **Perez, Pepe.** The Mandelbrot Set. *Mente Enjambre.* [Elektronski] 2012. http://www.mentenjambre.com/2012/12/el-conjunto-de-mandelbrot.html.

10. **Mandelbrot, Benoit.** *The fractal geometry of nature.* 1082.

11. **Wikipedia.** HostGator. *Wikipedia.* [Elektronski] 5. marec 2022. https://en.wikipedia.org/wiki/HostGator.

12. —. Web hosting service. *Wikipedia.* [Elektronski] 15. december 2021.

13. **Processing Foundation.** Setup. *p5js.* [Elektronski] 1. november 2013. https://p5js.org/reference/#/p5/setup.

14. —. angleMode. *p5.js.* [Elektronski] 1. november 2013. https://p5js.org/reference/#/p5/angleMode.

15. **Precessing Foundation.** push. *p5.js.* [Elektronski] 1. november 2013. https://p5js.org/reference/#/p5/push.

16. **Processing Foundation.** pop. *p5.js.* [Elektronski] 1. november 2013. https://p5js.org/reference/#/p5/pop.

17. **Ayodeji, Bolaji.** How to redirect HTTP to HTTPS Using .htaccess. *freecodecamp.* [Elektronski] 13. junij 2019. https://www.freecodecamp.org/news/how-to-redirect-http-to-https-using-htaccess/.

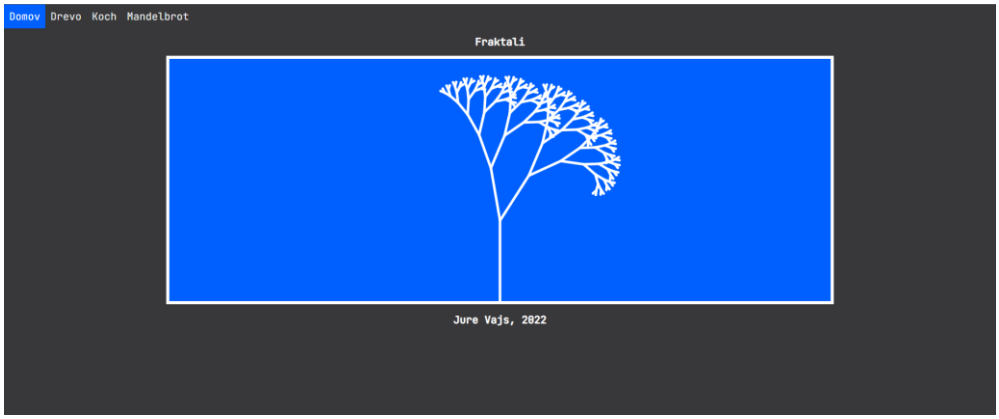
18. **Suberg.** Koch curve. *Code wars.* [Elektronski] 6. februar 2019. https://www.codewars.com/kata/5c5abf56052d1c0001b22ce5.

19. **Galowicz, Jacek.** C++17 STL Cookbook. *C++17 STL Cookbook.* s.l. : Packt, 2017.

7 STVARNO KAZALO

8 PRILOGE

Priloga 1 – Začetna stran

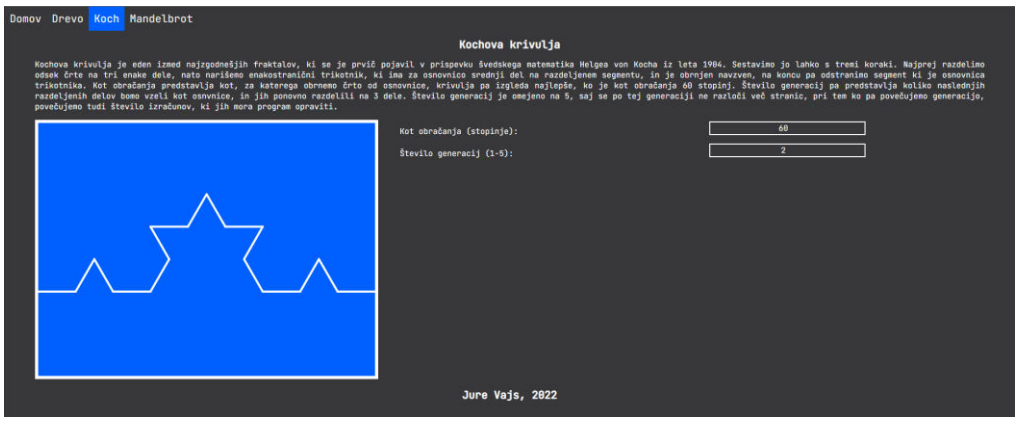


Slika 37: Začetna stran

Priloga 2 – Stran drevesa

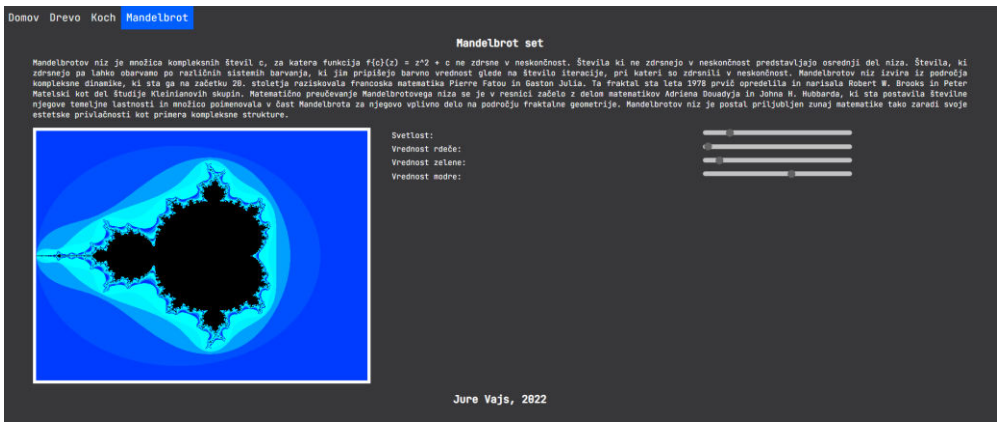
Slika 38: Stran drevesa

Priloga 3 – Stran Kochove krivulje



Slika 39: Stran Kochove krivulje

Priloga 4 – Stran Mandelbrotovega niza



Slika 40: Stran Mandelbrotovega niza

Priloga 5 – Povezava do kode za spletno stran

Fractals_website/Code at main · Burekinjo/Fractals_website (github.com)

Priloga 6 – Povezava do dokumentacije

Fractals_website/Dokumentacija at main · Burekinjo/Fractals_website (github.com)

Priloga 7 – Povezava do PowerPointa

Fractals_website/Powerpoint at main · Burekinjo/Fractals_website (github.com)