

SAPIENTIA ERDÉLYI MAGYAR TUDOMÁNYEGYETEM
MAROSVÁSÁRHELYI KAR,
INFORMATIKA SZAK



SAPIENTIA
ERDÉLYI MAGYAR
TUDOMÁNYEGYETEM

Dinamikus programozás vizualizátor

DIPLOMADOLGOZAT

Témavezető:
Dr. Kátai Zoltán,
Egyetemi docens

Végzős hallgató:
Stoica Attila

2023

UNIVERSITATEA SAPIENTIA DIN CLUJ-NAPOCA
FACULTATEA DE ȘTIINȚE TEHNICE ȘI UMANISTE,
SPECIALIZAREA INFORMATICĂ



UNIVERSITATEA
SAPIENTIA

Vizualizator pentru algoritmi de programare dinamică

LUCRARE DE DIPLOMĂ

Coordonator științific:
Dr. Kátai Zoltán,
Conferențiar universitar

Absolvent:
Stoica Attila

2023

**SAPIENTIA HUNGARIAN UNIVERSITY OF
TRANSYLVANIA
FACULTY OF TECHNICAL AND HUMAN SCIENCES
COMPUTER SCIENCE SPECIALIZATION**



SAPIENTIA
HUNGARIAN UNIVERSITY
OF TRANSYLVANIA

Visualizer for dynamic programming algorithms

BACHELOR THESIS

Scientific advisor:
Dr. Káta Zoltán,
Associate professor

Student:
Stoica Attila

2023

UNIVERSITATEA „SAPIENTIA” din CLUJ-NAPOCA
Facultatea de Științe Tehnice și Umaniste din Târgu Mureș
Programul de studii: Informatică

Viza facultății:

LUCRARE DE DIPLOMĂ

Coordonator științific: dr. Kátai Zoltán

Candidat: Stoica Attila
Anul absolvirii: 2023

a) Tema lucrării de licență:

Vizualizator pentru algoritmi de programare dinamică

b) Problemele principale tratate:

Studierea metodei de programare dinamică

Clasificarea algoritmilor de programare dinamică

Analizarea instrumentelor de vizualizare a algoritmilor de programare dinamică deja existente

Crearea unui soft-didactic propriu pentru algoritmilor de programare dinamică

c) Desene obligatorii:

Use-case, secvență, arhitectură

d) Softuri obligatorii:

Pagină web pentru vizualizarea algoritmilor de programare dinamică

e) Bibliografia recomandată:

Kátai Zoltán. "Algoritmusok felülnézetből". 2007. Scientia.

Meenakshi and Kamal Rawat. "Dynamic programming for coding interviews, a bottom-up approach to problem solving". 2017. Notion Press.

f) Termene obligatorii de consultații: lunar

g) Locul și durata practicii: Universitatea „Sapientia” din Cluj-Napoca,

Facultatea de Științe Tehnice și Umaniste din Târgu Mureș

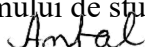
Primit tema la data de: 1. octombrie 2022.

Termen de predare: 2. iulie 2023.

Semnătura Director Departament



Semnătura responsabilului
programului de studiu



Semnătura coordonatorului



Semnătura candidatului



Declarație

Subsemnatul/a STOICA ATTILA, absolvent(ă) al/a specializării
INFORMATICA, promoția 19-23, cunoscând
prevederile Legii Educației Naționale 1/2011 și a Codului de etică și deontologie profesională a
Universității Sapientia cu privire la furt intelectual declar pe propria răspundere că prezenta
lucrare de licență/proiect de diplomă/disertație se bazează pe activitatea personală,
cercetarea/proiectarea este efectuată de mine, informațiile și datele preluate din literatura de
specialitate sunt citate în mod corespunzător.

Localitatea,

Data: CORUNCA

2023.06.13

Absolvent

Semnătura.....

Kivonat

A dolgozatom témája a dinamikus programozás algoritmusának vizualizálása egy könnyen alkalmazható felületen keresztül. A dinamikus programozás egy hatékony és kreatív módszer a problémamegoldásra. Az alapja az optimális megoldások tárolása és újrafelhasználása. Kicsit olyan, mint egy memóriajáték, ahol a korábbi lépéseket kihasználva könnyebben megtalálhatod a megoldást. Ezáltal időt és erőforrásokat takarít meg. Használják többek között optimalizációban, számítógépes grafikában és mesterséges intelligenciában is.

A dinamikus programozásnak az érthető bemutatása rendkívül fontos a diákok számára. Képzeljük el, hogy egy izgalmas kalandozásba hívjuk őket a problémamegoldás világába. Megmutatjuk nekik, hogyan törhetik le a bonyolult feladatokat egyszerűbb részproblémákra, majd ezeket könnyebben megoldva eljuthatnak a teljes megoldáshoz. Ehhez példákkal és gyakorlati alkalmazásokkal színesítsük a leckét, lehetővé téve számukra, hogy lássák, hogyan segítheti őket a dinamikus programozás az élet különböző területein, például adatelemzésben vagy videojáték-fejlesztésben. Egyedi interaktív feladatokkal és kreatív megközelítésekkel felkeltsük a diákok érdeklődését és készítjük őket a kritikus gondolkodásra és problémamegoldásra. Ezáltal fejlődik a kreativitásuk, a logikai gondolkodásuk és a csapatmunkára való képességük is.

Dolgozatom által az a célom, hogy minél érthetőbben és egyedibben mutassam be a dinamikus programozást, ezáltal keltsem fel a diákok figyelmét és megalapozzam az informatikai készségeiket számukra a jövőben. Mindezt egy weboldalon keresztül törekedtem megvalósítani, legfőképpen azért mert ez így bárki számára könnyen elérhető és alkalmazható.

Végül nagyon fontos tényező volt, hogy a vizualizálás olyan feladatok által történjen, melyek tökéletesen tükrözik az algoritmus alapvető tulajdonságait, mint a részproblémára való tagolás és a memoizálás. Továbbá felhasználó barát felület létrehozására törekedtem, ahol a vizualizáció sebessége is beállítható.

Rezumat

Tema lucrării mele este vizualizarea algoritmului de programare dinamică printr-o interfață ușor de utilizat. Programarea dinamică este o metodă eficientă și creativă de rezolvare a problemelor. Se bazează pe stocarea și reutilizarea soluțiilor optime. Este puțin asemănătoare unui joc de memorie, unde prin folosirea pașilor anteriori poți găsi mai ușor soluția. Astfel, economisești timp și resurse. Este folosită în optimizare, grafică computerizată și inteligență artificială.

Prezentarea clară a programării dinamice este extrem de importantă pentru elevi. Să ne închipuim că îi invităm într-o aventură captivantă în lumea rezolvării problemelor. Le arătăm cum pot descompune sarcinile complexe în probleme mai simple, pe care le pot rezolva mai ușor, astfel ajungând la soluția completă. Împreună cu exemple și aplicații practice, le arătăm cum programarea dinamică îi poate ajuta în diverse domenii ale vieții, cum ar fi analiza datelor sau dezvoltarea de jocuri video. Prin intermediul sarcinilor interactive și a abordărilor creative, stârnim interesul elevilor și îi îndemnăm să gândească critic și să rezolve probleme. Astfel, își dezvoltă creativitatea, gândirea logică și abilitățile de lucru în echipă.

Scopul lucrării mele este să prezint programarea dinamică cât mai clar și mai original posibil, pentru a capta atenția elevilor și a le asigura fundamentul abilităților IT pentru viitor. Am încercat să realizez acest lucru printr-un site web, deoarece este accesibil și ușor de utilizat pentru oricine.

A fost un factor foarte important ca vizualizarea să se facă prin intermediul unor sarcini care reflectă perfect caracteristicile fundamentale ale algoritmului, cum ar fi dezmembrarea în subprobleme și memoizarea. De asemenea, am încercat să creez o interfață prietenoasă pentru utilizatori, unde viteza de vizualizare poate fi ajustată.

Abstract

The topic of my paper is visualizing the algorithm of dynamic programming through an easily applicable interface. Dynamic programming is an efficient and creative method for problem-solving. Its foundation lies in storing and reusing optimal solutions. It's a bit like a memory game, where by leveraging previous steps, you can more easily find the solution. This saves time and resources. It is used in optimization, computer graphics, and artificial intelligence.

The clear presentation of dynamic programming is extremely important for students. Imagine inviting them on an exciting adventure into the world of problem-solving. Show them how they can break down complex tasks into simpler subproblems, and by solving these, they can eventually reach the complete solution. Enrich the lesson with examples and practical applications, allowing them to see how dynamic programming can assist them in various areas of life, such as data analysis or video game development. By incorporating unique interactive exercises and creative approaches, you pique students' interest and encourage critical thinking and problem-solving skills. This fosters the development of their creativity, logical thinking, and ability to work in teams.

The goal of my paper is to present dynamic programming as comprehensively and uniquely as possible, thereby capturing students' attention and laying the foundation for their future IT skills. I aimed to achieve this through a website, primarily because it is easily accessible and applicable to anyone.

Furthermore, it was crucial for the visualization to be carried out through tasks that perfectly reflect the fundamental properties of the algorithm, such as dividing problems into subproblems and memoization. Additionally, I strived to create a user-friendly interface where the visualization speed can also be adjusted.

Tartalomjegyzék

1. Bevezető	10
1.1. Mérföldkövek	12
2. Irodalmi tanulmányok	13
3. A rendszer specifikációi és architektúrája	15
3.1. Felhasználói követelmények	15
3.2. Rendszer követelmények	16
3.2.1. Funkcionális követelmények	16
3.2.2. Nem funkcionális követelmények	16
3.3. Architektúra	17
4. Használt programok és technológiák	19
4.1. Visual Studio Code	19
4.2. HTML	20
4.3. CSS	21
4.4. REACT	22
4.5. TYPESCRIPT	23
4.6. MUI	23
4.7. GITHUB	24
5. Projekt kivitelezése	25
5.1. Felület létrehozása	25
5.2. Működés	28
5.3. Kiválasztott feladatok	29
5.3.1. Minimum/Maximum költségű út megállapítása	29
5.3.2. Fibonacci számok	31
5.3.3. Catalan számok	32
5.3.4. Pénzértékes feladat(Mátrixos megoldás)	33
Összefoglaló	35
Köszönetnyilvánítás	37
Ábrák jegyzéke	38
Irodalomjegyzék	39

1. fejezet

Bevezető

A dinamikus programozás rendkívül hasznos módszer a problémamegoldás terén. Ez egy kreatív megközelítés, amelynek célja az optimális megoldások megtalálása és újrafelhasználása. A dinamikus programozás lehetővé teszi, hogy bonyolult problémákat felosszunk kisebb részproblémákra, amelyeket könnyebben megoldhatunk. Ezáltal időt és erőforrásokat takarítunk meg. Az alkalmazása széleskörű, és számos területen előnyöket nyújt.

Használják az optimalizációban, számítógépes grafikában, mesterséges intelligenciában és még sok más területen. Az egyedisége abban rejlik, hogy a problémák specifikus jellegétől függően kreatív megoldásokat kínál. A dinamikus programozás révén felismerhetünk rejtett mintázatokat és optimalizálhatjuk a folyamatokat. Ezáltal javul a hatékonyság és a teljesítmény. A dinamikus programozás kiemelkedő módszer a fejlesztők és a problémamegoldók számára, akik hatékony és innovatív megoldásokat keresnek a komplex problémákra.

A dolgozat az algoritmus edukatív jellegű bemutatására fekteti a hangsúlyt. A cél a feladatok lépésről lépésre való megoldása által a fő tulajdonságok érthető szemléltetése.

A dinamikus programozás olyan módszer, amely nemcsak az informatikai területen, hanem a mindennapi életben is kiválóan alkalmazható. Ez az algoritmusok és a problémamegoldás hatékony eszköze, amely lehetővé teszi, hogy a bonyolult feladatokat egyszerűbb részekre bontsuk, és ezáltal könnyebben kezelhetővé váljanak.

Gondoljunk például egy olyan helyzetre, amikor egy hosszú listánk van a teendő feladatainkkal. Az első reakciónk az lehet, hogy kezdjük előlről és egyenként végrehajtjuk a feladatokat. Azonban a dinamikus programozás megközelítésével, kategorizálhatjuk a feladatokat és meghatározhatjuk a leoptimálisabb sorrendet, amely minimalizálja az időt és az erőfeszítést. Ezáltal hatékonyabban tudjuk elvégezni a feladatainkat.

Egy másik példa az időbeosztásra vonatkozik. A dinamikus programozás segítségével hatékonyan tervezhetjük meg a napjainkat vagy a heti tevékenységeinket. Az algoritmus segít abban, hogy optimalizáljuk a tevékenységek sorrendjét és időtartamát, hogy a lehető legtöbbet érjük el az adott idő alatt. Ezáltal javíthatjuk a produktivitást és csökkenthetjük a stresszt.

Emellett a dinamikus programozás hasznos lehet a költségvetés tervezésében is. Például, ha tervezel egy utazást, ahol több helyet szeretnél meglátogatni, az algoritmus segíthet optimalizálni az útvonalat, hogy minimalizálja az utazási költségeket és időt.

Meenakshi és Kamal Rawat a következőt állítja a dinamikus programozásról: [2] A DP egy alulról felfelé haladó megközelítés a problémamegoldáshoz, ahol egy részproblémát csak egyszer van megoldva. Legtöbb esetben ez a megközelítés ellentmond az intuíciónak. Lehet, hogy megváltoztatást igényel a probléma megközelítésében. Még tapasztalt programozóknak is nehézséget okozhat DP problémák megoldása. ("DP is a bottom-up approach to problem solving where one subproblem is solved only once. In most cases this approach is counter-intuitive. It may require a change in the way we approach a problem. Even experienced coders struggle to solve DP problems")

A dinamikus programozás elvei széles körben alkalmazhatók az üzleti világban is. A vállalatok gyakran szembesülnek olyan problémákkal, mint a raktárkészletek optimalizálása vagy a termelési folyamatok hatékonyságának növelése. A dinamikus programozás segíthet ezeknek a problémáknak az elemzésében és optimalizálásában, így csökkentve a költségeket és növelve a profitot.

Dimitri P. Bertsekas az egyik legismertebb és legbefolyásosabb személy a dinamikus programozás és az optimális vezérlés területén a következőt említi: [3] A dinamikus programozás fontos eszköz mind a meghatározott, mind a sztochasztikus irányításban. ("Dynamic programming is an important tool in both deterministic and stochastic control.")

A további fejezetekben a weboldal létrehozási fázisai lesznek részletesen bemutatva.

1.1. Mérföldkövek

- A weboldal előkészítése
- Eredeti logo megtervezése
- A vizualizáció sebességének implementálása
- A legalkalmasabb feladattípusok kiválasztása
- A feladatok megoldásának kidolgozása
- Feladat lista létrehozása
- A felhasználó paraméter bevételének kidolgozása
- A paramétereknek megfelelő táblázat(ok) felépítése
- Lépésenkénti vizualizálás
- Bonyolultabb műveletek esetében számolás magyarázat
- Eredmény kimutatás
- Eredeti állapot visszahozásának lehetősége

2. fejezet

Irodalmi tanulmányok

A Meenakshi és Kamal Rawat által alkotott "Dynamic Programming for Coding Interviews, a Bottom-Up Approach to Problem Solving" című könyv kifejezetten a dinamikus programozás alkalmazására összpontosít a kódolási interjúk során. A szerzők részletesen bemutatják a dinamikus programozás algoritmusait és technikáit, és gyakorlati példákon keresztül mutatják be azok használatát a problémamegoldásban.

Az elsődleges előnye ennek a könyvnek az volt, hogy kifejezetten a kódolási interjúkra fókuszált. A könyv segítséget nyújtott a dinamikus programozás alapelveinek és technikáinak megértésében, amelyeket gyakran alkalmaznak ilyen típusú interjúkon. A kódolási interjúk kihívást jelenthetnek a programozók számára, és a dinamikus programozás gyakran megjelenik ezeken a beszélgetéseken. A könyv által bemutatott bottom-up megközelítés és az algoritmusok lépéseinek részletes magyarázata jelentős segítséget nyújtott abban, hogy hatékonyan tudjam alkalmazni a dinamikus programozást a weboldal tervezésében.

A könyvben található példák és gyakorlati feladatok rendkívül hasznosak voltak, mivel konkrét esetekben mutatták be a dinamikus programozás alkalmazását. Ez lehetővé tette számomra, hogy valós problémákat és adathalmazokat vizsgáljak, és ezekre alkalmazzam a dinamikus programozás technikáit. A gyakorlati példák segítségével megtanulhattam a lépéseket, amelyeket a dinamikus programozás során követnem kell, és megértettem az algoritmusok működését és hatékonyságát.

A bemutatott bottom-up megközelítés különösen előnyös volt a weboldal elkészítése során. A bottom-up megközelítés lépcsről lépésre halad a probléma megoldása felé, kezdve a legkisebb részfeladatokról és fokozatosan haladva a teljes feladat megoldása felé. Ez a módszer lehetővé tette számomra, hogy a weboldal dinamikus programozási algoritmusait lépcsről lépésre vizualizáljam, és interaktív módon mutassam be a megoldási folyamatot. A könyv által nyújtott részletes leírások és magyarázatok segítettek abban, hogy érthetően és hatékonyan implementáljam ezeket az algoritmusokat a weboldal kódzába.

A "Dynamic Programming and Optimal Control" könyv Dimitri P. Bertsekas szerzőtől részletesen foglalkozik a dinamikus programozás és az optimális vezérlés elméletével. Bemutatja a dinamikus programozás algoritmusainak elméleti alapjait és gyakorlati alkalmazásait. A szerző részletesen elemzi az optimális vezérlési problémák megoldására alkalmazott dinamikus programozási technikákat.

Az [4] Anany Levitin által írt "Introduction to the Design and Analysis of Algorithms" és a [5] Robert Sedgewick és Kevin Wayne által írt "Algorithms" című könyvek részletesen foglalkoznak a dinamikus programozás technikájával, példákkal és gyakorlati alkalmazásokkal, valamint tárgyalják a feladatok felbontásának és a megoldások optimális kiválasztásának fontosságát.

Továbbiakban nagyon hasznosnak bizonyult [1] Kátai Zoltán "Algoritmusok felületéből" című alkotása, melyben részletesen megfigyelhető a bemutatott algoritmusok hatékonysága és időkomplexitása. A rekurzió és dinamikus programozás közötti hatékonysági különbségek megértésében kiváltképpen előnyös forrás volt.

Ezen irodalmi források és információk alapján a dinamikus programozás vizualizálásához kialakítottam egy hatékony és interaktív weboldalt. Az irodalom felhasználása segített megérteni a dinamikus programozás elméletét, különböző algoritmusokat és módszereket, valamint megismerni a már létező megoldásokat és alkalmazásokat. Az így szerzett tudás lehetővé tette a weboldal hatékony tervezését és fejlesztését, amely lehetővé teszi a felhasználók számára a dinamikus programozás algoritmusainak és lépéseinek vizualizálását, megértését és interaktív tanulását.

3. fejezet

A rendszer specifikációi és architektúrája

3.1. Felhasználói követelmények

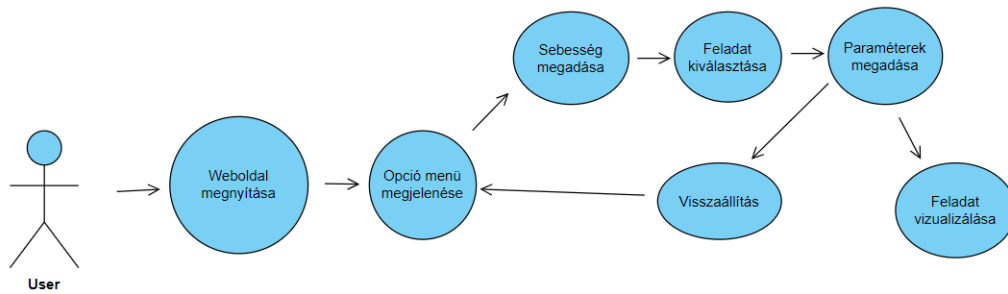
A felhasználó első lépésben megnyitja a weboldalt. Az oldal betöltése után a jobb oldalon megjelenik egy opció menü, amelyben különböző feladatok közül lehet választani, melyeket szeretnénk vizualizálni. Emellett egy input mező is található, ahol meg lehet adni a vizualizáció sebességét másodpercekben. Fontos megjegyezni, hogy amíg a sebesség paramétert nem adta meg a felhasználó, addig nem lehetséges feladatot kiválasztani.

Miután a felhasználó megadta a sebesség értékét és kiválasztott egy feladatot, az oldalon megjelenik egy szövegdoz, amely leírja, hogy milyen további paramétereket kell megadni a kiválasztott feladattól függően. Például, ha a feladat egy két dimenziós tömb, akkor a felhasználónak meg kell adnia a mátrix méreteit. Ezután soronként, szóközzel elválasztva, meg kell adnia az értékeket.

Miután az összes érték megadásra került, megjelenik a feladathoz kapcsolódó egy vagy két dimenziós tömb vagy tömbök, valamint egy lépések magyarázatát tartalmazó szövegdoz. A lépések magyarázó szövegdozban részletesen le vannak írva a bonyolultabb műveletek, amelyek a megoldás folyamatát lépésről lépésre mutatják be.

Emellett a felhasználó számára elérhető lesz egy visszaállítás gomb is, amely lehetővé teszi a felhasználó számára, hogy újramezdje az egész folyamatot.

Ezzel a vizualizáció elkezdődik, és a cellák színezése által megfigyelhető lesz a dinamikus megoldás folyamata lépésről lépésre. A színek segítségével könnyen követhető lesz, hogy melyik cella változott és hogyan változott az egyes lépések során. Ezáltal a felhasználó jobban megértheti a vizualizált feladat megoldását.



3.1. ábra. Use Case

3.2. Rendszer követelmények

3.2.1. Funkcionális követelmények

A dinamikus programozást vizualizáló weboldalhoz a felhasználóknak egy olyan eszközre van szükségük, amelyen fut egy böngésző. Ez lehet akár számítógép, laptop, tablet vagy okostelefon.

A felhasználóknak internetkapcsolatuknak kell lennie, hogy elérjék és használják a weboldalt. Az internetkapcsolat lehet vezetékes vagy vezeték nélküli, de stabil és megbízható kell legyen annak érdekében, hogy a weboldal zavartalanul működjön.

A böngésző szintén fontos a weboldal használatához. A legtöbb böngésző támogatja a dinamikus programozást és a webes technológiákat, mint például a HTML, CSS és JavaScript. Ajánlott a legfrissebb verzióval rendelkező böngésző használata annak érdekében, hogy a weboldal optimálisan működjön, és a legújabb funkciókat támogassa.

3.2.2. Nem funkcionális követelmények

A weboldalnak reszponzívnak és adaptív módon kell működnie különböző készülékeken és kijelzőméreteken. Legyen az asztali számítógép, laptop, tablet vagy okostelefon, a weboldalnak jól kell működnie és megfelelően kell megjelenítenie a tartalmat a különböző eszközökön.

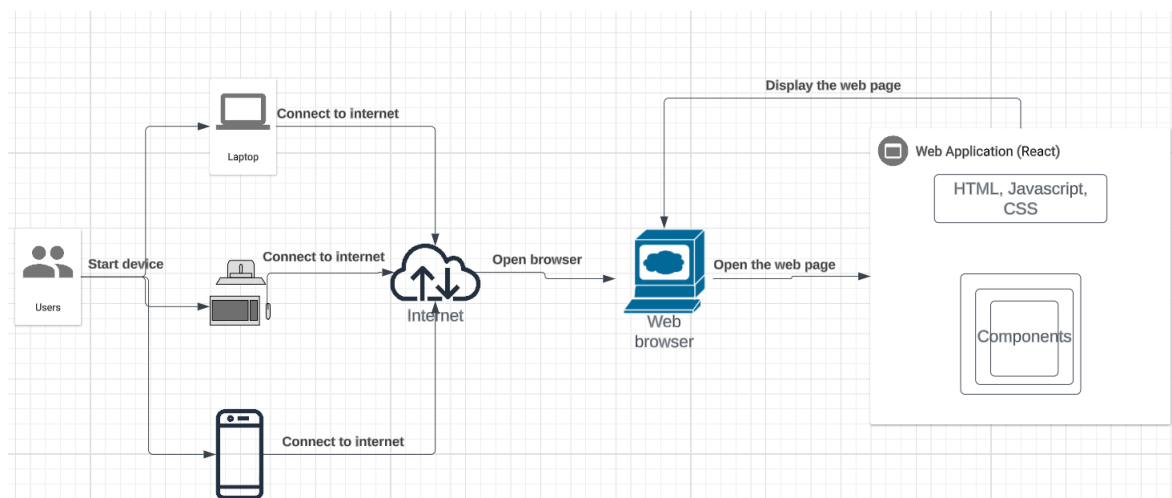
Fontos a keresztböngésző kompatibilitás is. A weboldalnak megfelelően kell működnie a különböző böngészőkben, mint például a Chrome, Firefox, Safari vagy Edge. A vizualizáció és a funkcionalitás ugyanúgy működjön minden támogatott böngészőben anélkül, hogy jelentős különbségek lennének.

3.3. Architektúra

Alább megtekinthető a 3.2-es ábrán az alkalmazásom architektúra diagramja. Első sorban, az applikációt dinamikus programozást kedvelő személyek, azon felül diákok illetve tanárok vehetik használatba, hogy jobban elmerülhessenek a dinamikus programozás rejtelmeiben. Viszont azon felül bárki számára elérhető, nyitott az oldal aki akár egy kicsit is megszeretne ismerkedni akár ezzel az algoritmussal, de akár a programozással is.

Felhasználóként nincs szükségünk csak egy technológia által nyújtott eszközre, mint a számítógép, laptop, okostelefon vagy ezekhez hasonló digitális eszközök. Mindezek mellett fontos megemlíteni, hogy szükség van valamilyen hálózatra, mint például router vagy internet. Ennek eredményében elérhetővé válik kereső programok, ahol rá lehet keresni a weboldalra. Ebből adódóan megnyílik a React fejlesztői környezetben keresztül létrehozott felület. A projekt több komponensből áll, hasonlóan az osztályokhoz, amik egy adott részfeladatot megoldanak, majd ezek alkotják az alkalmazást. A komponensek egymás között props-okkal kommunikálnak, egymásba meghívhatóak bizonyos kritériumok alapján, ugyanakkor újra felhasználhatóak ami egy remek előnyt nyújt egy webfejlesztő számára. Plusz a komponensek által a projekt struktúrája rendezetebb, átláthatóbb formába kerül.

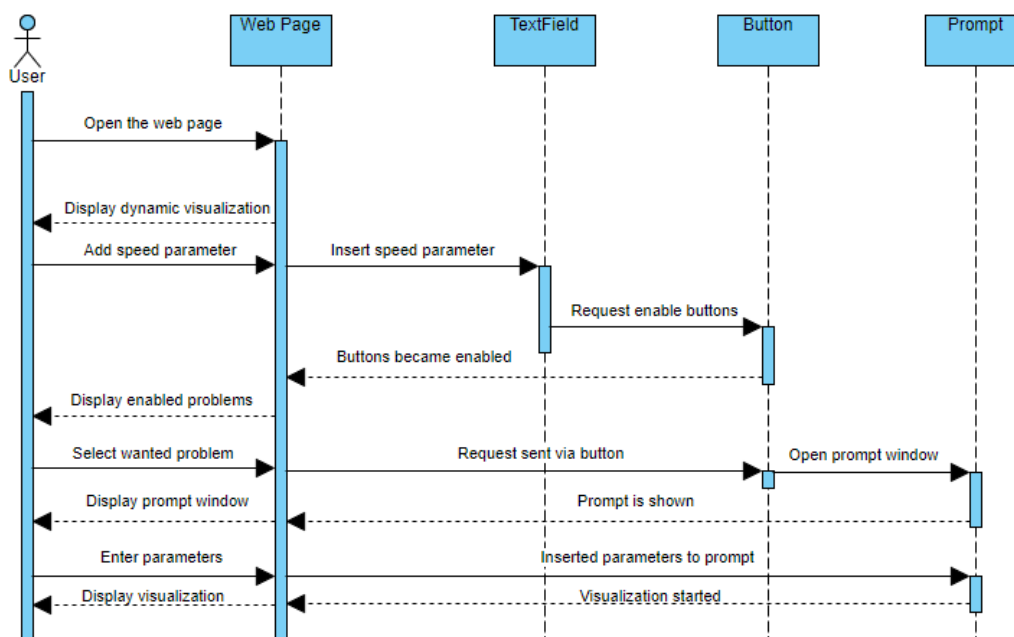
Mivel hálózatra van szükségünk ezért alapértelmezetten ha nincsen áram, nincs internet, ennek okán nem működne az oldal. Ilyen fajta hiba során várni kell, viszont más rendszerhiba alkalmával az adminisztrátor végzi el a helyreállítást.



3.2. ábra. Architektúra diagram

A szekvencia diagramom amit lentebb a 3.3.-as ábra reprezentál, a felhasználó és a weboldal közötti időbeli interakciója látható. Az a folyamat amely során a weboldal megnyitásától egészen a kiválasztott feladat vizualizálásának megtekintéséig eljut.

Első sorban, a megadott link által megnyitja az oldalt, így megjelenik a dinamikus programozás vizualizálásáról szóló felület. Mindezek után a felhasználó első teendője, hogy kiválasszon egy sebességet. Amint ez a sebesség ki volt választva jelzés küldődik a problémákat reprezentáló gombok felé, ezáltal pedig aktiválva lesznek. Tehát, elérhetővé válnak a feladatok amik közül választhat a felhasználó, hogy megszeretné nézni a vizualizálását, működését dinamikus programozás szintjén. Ezt követően felugrik egy prompt ablak ami az adott feladathoz szükséges paramétereket kéri be. (Ez lehet akár egy paraméter mely a szekvencia hosszát vagy több paraméter mely a két dimenziós tömb méretét és ennek elemeit jelöli). Minek után a felület felhasználója megadta a bemeneteket a weboldal háttere a megfelelő lépéseket elvégezve elindítja számára a feladat kirajzolását, vizualizációját, hogy hogyan is működik dinamikus programozás algoritmus felhasználásával a kért probléma, hogy egyszerűbben és könnyebben érthetővé váljon.



3.3. ábra. Szekvencia diagram

4. fejezet

Használt programok és technológiák

4.1. Visual Studio Code

A Visual Studio Code (VS Code) az egyik leginkább kifinomult és sokoldalú kódszerkesztő a piacon, amelyet sok programozó előszeretettel használ. Számos különleges tulajdonsága miatt döntöttem úgy, hogy ezt az eszközt választom a dinamikus programozást vizualizáló weboldalam elkészítéséhez.

A VS Code rendkívül könnyű és intuitív kezelőfelülettel rendelkezik. A letisztult és felhasználóbarát dizájn lehetővé teszi a gyors navigációt és a kényelmes munkavégzést. A személyre szabható felület segítségével könnyedén testre szabhatom a megjelenést és a funkciókat az egyéni igényeimnek megfelelően.

Rengeteg kiegészítő érhető el hozzá, amelyek segítségével kényelmesen dolgozhatok különböző programozási nyelveken és keretrendszeren. Az integrált hibakereső és a kódszervezési eszközök könnyebbé teszik a hibák felderítését és a kód optimalizálását. Emellett a Git integrációja lehetővé teszi a könnyű verziókezelést.

A VS Code kiemelkedő teljesítményt nyújt. Az alacsony erőforrás-igénye miatt zökkenőmentesen fut akár régebbi hardvereken is. Ezen felül, a beépített feladatkezelő rendszer lehetővé teszi a hatékony munkavégzést a feladatok és munkamenetek könnyű váltásával.

Továbbá ingyenes és nyílt forráskódú. Ez azt jelenti, hogy bárki szabadon letöltheti és használhatja anélkül, hogy fizetnie kellene érte. Ez nagyon előnyös, különösen azért, mert a fejlesztőközösség folyamatosan dolgozik az új funkciók és javítások bevezetésén, ami még jobbra teszi az alkalmazást.

Az említett okok miatt tehát a Visual Studio Code-ot választottam a dinamikus programozást vizualizáló weboldalam fejlesztéséhez. A könnyű kezelőfelülete, a bővíthetősége, a kiemelkedő teljesítménye és az ingyenessége mind-mind hozzájárultak ahhoz, hogy hatékonyan és élvezetesen dolgozhassak a projekten.

4.2. HTML

A HTML (HyperText Markup Language) egy alapvető webes jelölőnyelv amely lehetővé teszi a tartalom strukturálását és megjelenítését a böngészőkben.

Rendkívül egyszerű és könnyen érthető nyelv, ami azt jelenti, hogy gyorsan elsajátítható és könnyen kezelhető. Az alapvető HTML elemei, mint például a fejléc, a bekezdések, a listák és a táblázatok, egyszerűen megérthetők és könnyen felhasználhatók. Ezáltal könnyedén tudtam strukturálni és formázni a weboldalam tartalmát, ami segítette a felhasználóbarát felület kialakítását.

```
You, 5 months ago | 1 author (You)
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="utf-8" />
  <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
  <meta name="viewport" content="width=device-width, initial-scale=1" />
  <meta name="theme-color" content="#000000" />
  <meta name="description" content="Web site created using create-react-app" />
  <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
  <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
  <title>Dynamic Programming Visualization</title>
</head>

<body>
  <noscript>You need to enable JavaScript to run this app.</noscript>
  <div id="root"></div>
</body>

</html>      You, 5 months ago • Created new project logo ...
```

4.1. ábra. Projekt html struktúrája

A HTML egy platformfüggetlen nyelv, ami azt jelenti, hogy a weboldalam elérhető és működőképes különböző operációs rendszereken és eszközökön. Ezáltal a weboldalam széles körben hozzáférhető és kompatibilis a felhasználók számára, akik különböző eszközöket és böngészőket használnak.

4.3. CSS

A CSS (Cascading Style Sheets) nyelv lehetővé teszi a weboldal megjelenésének testreszabását és stílusának meghatározását.

A CSS rendkívül hatékony eszköz a weboldalak stílusának kezeléséhez. A HTML-hez képest a CSS lehetővé teszi a pontosabb és részletesebb megjelenési tulajdonságok beállítását. Például a betűtípusok, a színek, a méretek, a margók és a keretek könnyedén testreszabhatók a CSS segítségével. Ez lehetővé teszi a weboldalam különleges és vonzó megjelenésének létrehozását, ami segít a felhasználók figyelmének felkeltésében és a vizualizáció hatékony közvetítésében.

Alkalmazásával lehetőségem nyílt az elemek elrendezésének és elhelyezésének pontos irányítására. Könnyedén hozhatok létre reszponzív elrendezéseket, amelyek alkalmazkodnak a különböző képernyőméretekhez és eszközökhöz. Ez különösen fontos a dinamikus programozást vizualizáló weboldal esetében, mert így biztosíthatom, hogy a vizualizációk megfelelően jelenjenek meg és könnyen áttekinthetők legyenek.

A CSS támogatja az animációkat és az átmeneteket, amelyek segítségével életet lehelhetek a dinamikus programozást vizualizáló weboldalamba. Az animációs tulajdonságok lehetővé teszik a vizualizációk sima és vonzó mozgását, ami felhasználóbarát és látványos élményt nyújt a felhasználóknak. Az átmenetek segítségével pedig fokozatosan változtathatom a stílusokat vagy a tulajdonságokat, ami további vizuális érdekességeket adhat a weboldalnak.

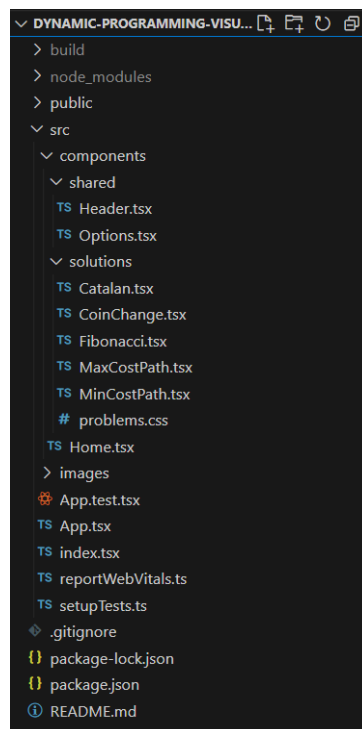
4.4. REACT

A dinamikus programozást vizualizáló weboldalam elkészítéséhez a React JavaScript keretrendszert választottam, mert ez a technológia lehetővé teszi a felhasználóbarát és interaktív felület kialakítását, valamint hatékonyan kezeli a dinamikus adatokat.

Rendkívül hatékony megoldást nyújt a felhasználói felületek kialakításához. A React komponens alapú architektúrája lehetővé teszi a weboldal moduláris felépítését, ahol minden részletet önálló komponensként kezelhetek. Ez nagyban megkönnyíti a kódolást, az újrafelhasználást és a karbantarthatóságot. A komponensek egyszerűen felépíthetők és testreszabhatók, és könnyedén kombinálhatók a weboldal különböző részeinek létrehozásához.

A React állapotkezelési mechanizmusa lehetővé teszi a dinamikus adatok követését és frissítését, aminek eredményeként az alkalmazásom gyorsan és hatékonyan reagál a felhasználói interakciókra. Az állapotkezelés segítségével a weboldalam folyamatosan frissül és a változó adatok automatikusan újra renderelődnek a felületen.

Rendelkezik egy erőteljes virtuális DOM (Document Object Model) algoritmussal. Ez azt jelenti, hogy a React csak a ténylegesen változott részeket frissíti a felhasználói felületen, és nem az egész oldalt. Ez lehetővé teszi a gyors és hatékony újra renderelést, ami nagyon előnyös a dinamikus programozást vizualizáló weboldal esetében, ahol a vizualizált elemek gyakran változnak és frissítést igényelnek.



4.2. ábra. Projekt struktúrája

Továbbá a React egyike a legnépszerűbb JavaScript keretrendszereknek, és egyre több fejlesztő és vállalat választja ezt a technológiát. Ez biztosítja a tartósságot, a közösségi támogatást és az új fejlesztések folyamatos elérhetőségét.

4.5. TYPESCRIPT

A TypeScript nyelvet azért választottam, mert ez a technológia lehetővé teszi a megbízhatóbb és könnyebben karbantartható kódbázis kialakítását.

A TypeScript egy szuperszett a JavaScript nyelvre, ami statikus típusellenőrzést tesz lehetővé. Ez azt jelenti, hogy a fejlesztés során már a kódírás közben észlelhetőek a potenciális hibák, például típushibák vagy hiányzó tulajdonságok. Ez segít elkerülni a futásidejű hibákat és növeli a kód minőségét. A statikus típusellenőrzés előnyös a nagyobb projektekben, mert segít a kód olvashatóságának és karbantarthatóságának javításában, valamint csökkenti a hibák lehetőségét.

Könnyen integrálható a modern fejlesztési folyamatokba. Támogatja a moduláris kódolást és a Build rendszereket, mint például a Webpack. Ez lehetővé teszi a kód optimalizálását és a fájlok méretének csökkentését a produkciós környezetben.

4.6. MUI

A MUI (Material-UI) komponenseket azért alkalmaztam, mert ezek a komponensek lehetővé teszik a gyors és esztétikus felhasználói felület kialakítását.

A komponensek kész és tesztelt megoldást nyújtanak a felhasználói felület építéséhez. Az előre elkészített komponensek, például gombok, űrlapelemek, navigációs sávok és táblázatok könnyen használhatók és testreszabhatók. Ez felgyorsítja a fejlesztést, mert nem kell az alapvető funkciókat és stílusokat újra létrehozni. A MUI komponensek konzisztens megjelenést és UI irányelveket követnek, ami egy egységes és professzionális kinézetet eredményez a weboldalon.

A MUI támogatja a React keretrendszert, így a React komponensekkel egyszerűen használhatók. Emellett számos kiegészítő könyvtár és bővítmény elérhető a MUI-hoz, amelyek további funkcionalitást és komponenseket kínálnak. Ez lehetővé teszi a széleskörű funkcionalitás és a speciális igények kielégítését.

4.7. GITHUB

A GitHubot választottam a dinamikus programozást vizualizáló weboldalam elkészítéséhez, mert ez a platform rendkívül hatékony eszközt biztosít a kódközpontú munkafolyamatok kezeléséhez és a verziókezeléshez.

A GitHub egy távoli tároló rendszer, amely lehetővé teszi a kód és más fájlok központi kezelését. A kódomat könnyedén feltölthetem és tárolhatom a GitHub tárolóban, így mindig biztonságban van és könnyen hozzáférhető a fejlesztési folyamat során. Emellett a GitHub által kínált verziókezelési rendszer lehetővé teszi, hogy nyomon kövessem a változtatásokat, összehasonlítsam a különböző verziókat és visszaállítsam korábbi változatokat, ha szükséges.

Mivel a nyílt forráskódú közösség támogatója lehetőségem van a projektet nyilvánosan elérhetővé tenni, így más fejlesztők szabadon hozzájárulhatnak a fejlődéséhez és a javításához. Ezáltal a projekt hosszú távon fenntarthatóvá válik és továbbfejleszthető a közösség erejével.

5. fejezet

Projekt kivitelezése

5.1. Felület létrehozása

Az elkészítési folyamat során a dinamikus programozást vizualizáló weboldalamhoz a React keretrendszert és a create-react-app eszközt választottam, hogy gyorsan és hatékonyan hozzáférhető és skálázható alkalmazást hozzak létre.

Először is, a create-react-app parancssoros eszközt használtam a React alkalmazásom inicializálásához. Ez az eszköz automatikusan konfigurálja a projektet a fejlesztéshez, beállítja a szükséges fájlokat és könyvtárakat, valamint optimalizálja a kód build folyamatát. Ez lehetővé tette számomra, hogy gyorsan elkezdjem a fejlesztést, anélkül hogy mélyrehatóan kellene foglalkoznom a konfigurációval.

Miután létrejött a projekt struktúrája, elkezdtem dolgozni a weboldal megjelenésén. Először is, létrehoztam egy egyedi logót, amely a weboldal sajátosságait tükrözi. A logó létrehozásához egy online grafikai szerkesztő eszközt használtam, a Canvat. A logó a weboldal arculatának fontos eleme volt, és hozzájárult a brand és az azonosítás erősítéséhez.



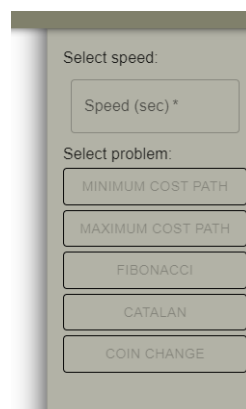
5.1. ábra. Dynamic Programming Visualization logo

A következő lépésben kialakítottam a weboldal fejléc részét, amely az AppBar komponensből, az Image komponensből és a Typography komponensből állt. Az AppBar komponens a MUI-ban szolgáltatott alapvető header komponens, amely tartalmazza a navigációs sávot és egyéb információkat. Az Image komponens segítségével beillesztettem a korábban létrehozott egyedi logót. A Typography komponens pedig a szöveges elemek formázására és megjelenítésére szolgált, például a weboldal címének vagy alcímének megjelenítésére.



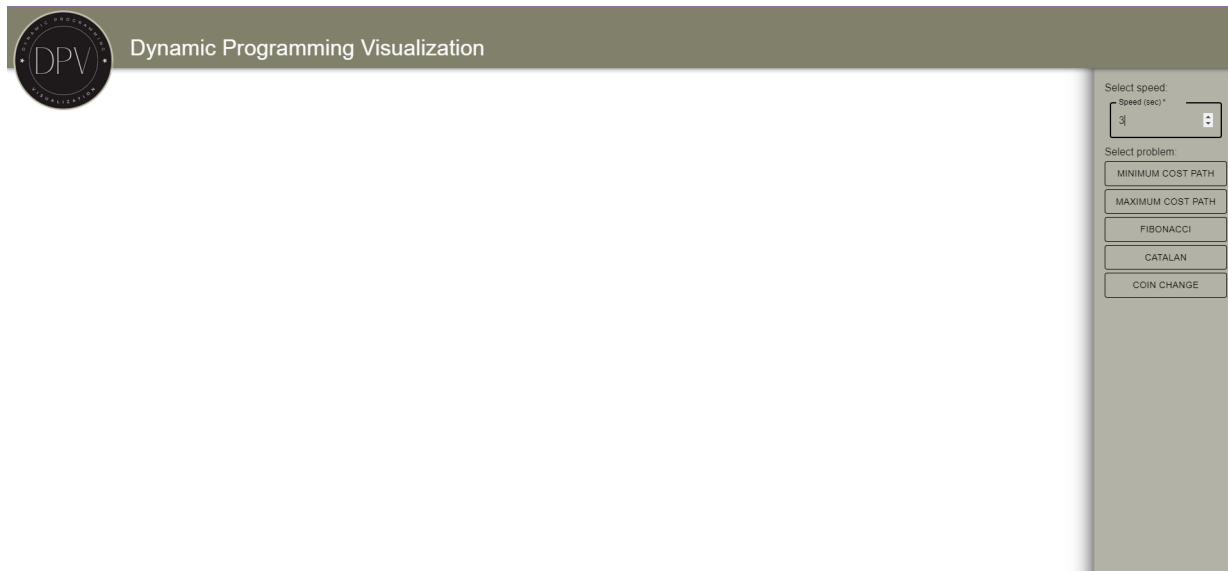
5.2. ábra. Fejléc

Ezután létrehoztam egy opció menüt a weboldalhoz. Az opció menü megjelenítéséhez és kezeléséhez a MUI-ban elérhető komponenseket használtam, mint például a Button, Divider, Drawer, TextField, Toolbar és Typography. A Button komponens segítségével elérhetővé tettem a feladatokat melyek közül a felhasználó választhat. A Divider komponens segítségével elkülönítettem menüben lévő elemeket a headertől. A Drawer komponens egy kihúzható fiókszerű menü, amely a weboldal esetében állandóan nyitott állapotban van és tartalmazza az opciókat. A TextField komponens segítségével lehetőséget adtam a felhasználónak a vizualizáció sebességének bevitelére másodpercekben. A Toolbar komponens a MUI által biztosított navigációs eszközök összessége, amelyek könnyű hozzáadást és testreszabást tesznek lehetővé. A Typography komponens pedig a szöveges tartalmak megjelenítésére és formázására szolgált az opció menüben.



5.3. ábra. Opció menü

Az említett komponenseket CSS segítségével stílizáltam, hogy egyedi megjelenést és összhangot érjek el. A CSS szabályok segítségével testreszabtam a komponensek háttérszíneit, betűtípusát, méretét és elrendezését. Az osztályok és szelektorok segítségével specifikus stílusokat adtam hozzájuk, hogy illeszkedjenek a weboldal általános dizájn-jegyeihez. A CSS stílusok lehetővé tették, hogy harmonikus és esztétikus megjelenést érjek el az opció menünél, az AppBar-nél, az Image-nél és a többi MUI komponensnél.



5.4. ábra. Page

Ezen lépések eredményeként a dinamikus programozást vizualizáló weboldalam egyedülálló, a React és a MUI komponensek segítségével létrehozott felhasználóbarát és esztétikus felülettel rendelkezik. A React biztosítja a dinamikus adatkezelést és a komponens alapú architektúrát, míg a MUI komponensek könnyen testreszabható és modern megjelenést nyújtanak. A kész weboldal magas szintű funkcionalitással rendelkezik, könnyen kezelhető, és felhasználói élmény szempontjából is kiemelkedő.

5.2. Működés

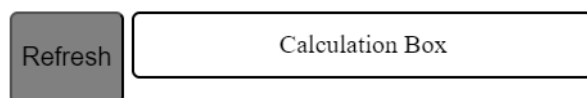
Amikor a felhasználó megnyitja a weboldalt, először láthatja az opció menüt, ahol be kell írnia a vizualizáció sebességét másodpercekben. Ez a sebesség szabályozza, hogy milyen gyorsan történik a vizualizáció. Ezt követően a felhasználó választhat egy feladatot a rendelkezésre álló lehetőségek közül.

A feladat kiválasztása után a felhasználótól bekérésre kerülnek a feladattól függő paraméterek. Ezek a paraméterek olyan specifikus adatok, amelyekre a feladat megoldásához szükség van. A felhasználó beírja ezeket az adatokat és tovább lép a következő lépésre.

A feladatokat külön komponensekben kezeltem le a React keretrendszerben. Mind-egyik feladatnak megvan a saját komponense, amely a megoldás lépéseit tartalmazza. Ezen komponensekben létrehoztam egy vagy két dimenziós táblázatokat, amelyek lépésről lépésre bemutatják a feladat megoldásának felépítését. A tömbökben tárolt adatok vizuálisan jelennek meg a weboldalon, így a felhasználó lépésről lépésre követheti a megoldás folyamatát.

A vizualizációhoz egyedi stílusokat alkalmaztam, amelyeket közös CSS fájlban tároltam. Ez az egyedi stílusbiztosítja a weboldal összehangolt és esztétikus megjelenését az összes feladat esetén. A közös CSS fájlban beállított stílusokat alkalmaztam a komponensekre, a háttérszínekre, a betűtípusokra és a méretekre, hogy a vizualizáció egységes és vonzó legyen a felhasználók számára.

Miután a felhasználó megadta a szükséges paramétereket és elindította a vizualizációt, a műveletek fokozatosan megjelennek a képernyőn. Bonyolultabb műveletek esetén egy szövegdobozban megjelenik a művelet, ami segít a felhasználónak követni az aktuális lépéseket. A felhasználó bármikor újra kezdheti a teljes folyamatot a refresh gombra kattintva, így lehetősége van többször is végrehajtani a vizualizációt a különböző paraméterekkel.



5.5. ábra. Refresh gomb és művelet magyarázó

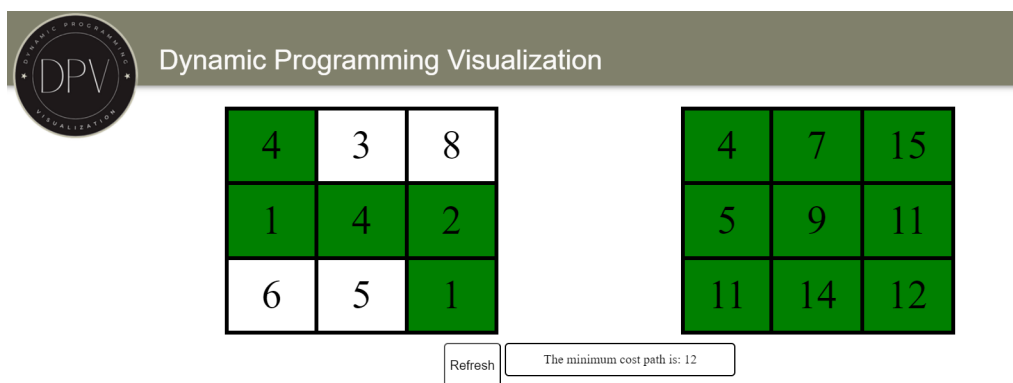
Ezáltal a dinamikus programozást vizualizáló weboldalam lehetőséget nyújt a felhasználóknak a programozási feladatok interaktív és érthető módon történő vizualizációjára. A React keretrendszer és a weboldalon alkalmazott egyedi funkciók és stílusok garantálják a felhasználók számára a kényelmes és megfelelő élményt a dinamikus programozás világában.

5.3. Kiválasztott feladatok

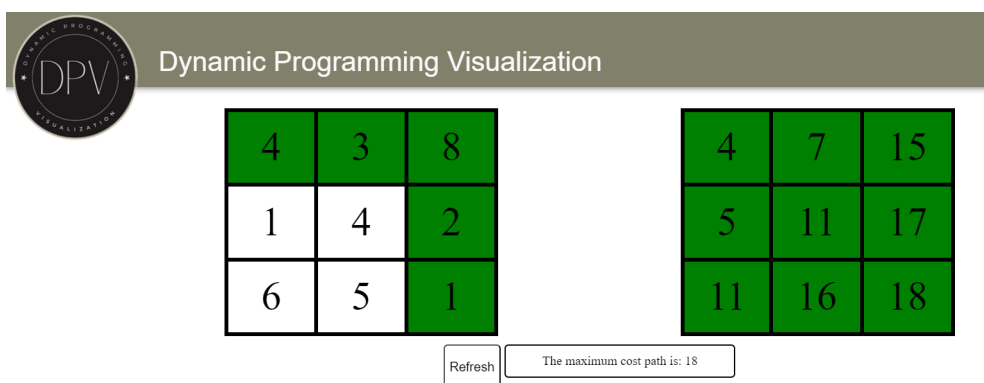
5.3.1. Minimum/Maximum költségű út megállapítása

A minimum/maximum út kiszámítása az egyik fontos probléma a dinamikus programozás területén. A dinamikus programozás algoritmusok segítségével hatékonyan és optimalizáltan lehet megoldást találni erre a problémára.

Ennek a feladatnak a megoldása során arra törekszünk, hogy megtaláljuk a legolcsóbb/legdrágább költségű utat egy adott tömbben két pont között. A megoldásban az út a felső bal oldali cellából indul és az jobb oldali alsó cellában van a célpont és átlósan nem lehet menni.



5.6. ábra. Minimum költségű út példa



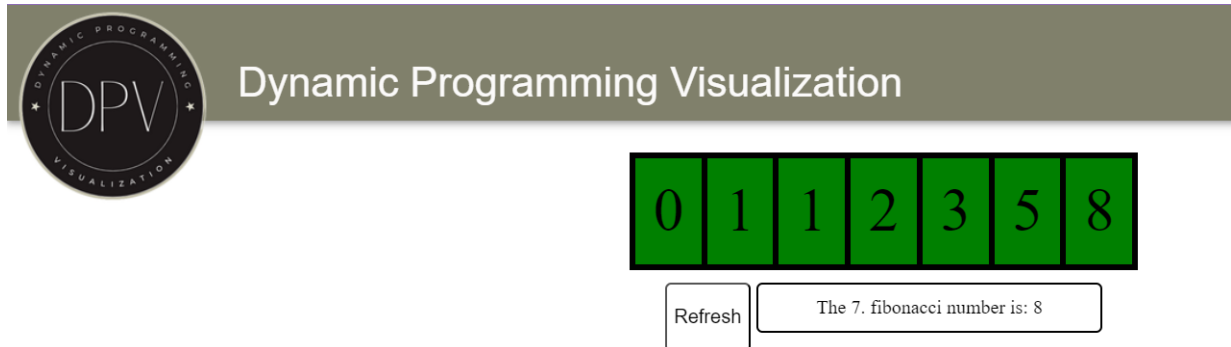
5.7. ábra. Maximum költségű út példa

A megoldásban létrehozom a mátrixot melyben dinamikus programozást alkalmazva feltöltök elemekkel. Ezt követően megkeresem a minimum/maximum költségű utat a mátrix segítségével és elmentem az eredményt, majd végrehajtok néhány animációt és módosítást a HTML elemeken, hogy megjelenítsem a minimális költségű utat a felhasználónak.

5.3.2. Fibonacci számok

A Fibonacci-sorozat egy olyan számsorozat, ahol az aktuális szám a két előző szám összege. Tehát az első két szám a sorozatban mindig 0 és 1, majd az újabb számokat a következőképpen számoljuk ki: az aktuális szám a két előző szám összege.

Amiután kiszámolunk egy Fibonacci-számot elmentjük azt a táblázatban vagy tömbben. Így minden újabb Fibonacci-számhoz csak egyszer kell számítást végezni, és a korábban kiszámolt értékekhez könnyen hozzáférhetünk.



5.8. ábra. Fibonacci számok

Az algoritmus futási ideje lineáris, mivel minden Fibonacci-szám kiszámításához csak egyszer kell hozzáférni a táblázatban vagy tömbben tárolt előző értékekhez.

Ez a dinamikus programozáson alapuló megoldás hatékonyabb, mint az egyszerű rekurziós megközelítés, mivel elkerüljük a felesleges ismétlődő számításokat.

Ezzel a módszerrel könnyedén és gyorsan kiszámíthatók a Fibonacci-számok, még nagyobb számsorozatok esetén is.

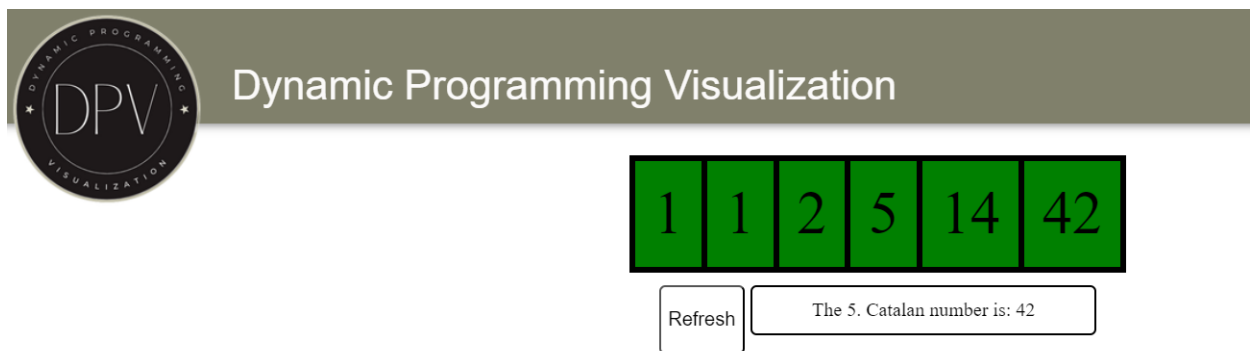
5.3.3. Catalan számok

A Catalan-számok kiszámítása dinamikus programozás segítségével hatékonyabbá tehető. Ehhez először létrehozunk egy táblázatot vagy tömböt, amelyben tároljuk az eddigi Catalan-számokat.

Kezdetben az első Catalan-szám mindig 1. Ezt elhelyezzük a táblázatunkban vagy tömbünkben.

Ezután iteratív módon haladunk előre a számsorozatban. Az aktuális Catalan-számot az előző Catalan-számok és az előző számok közötti kombinációk alapján számoljuk ki. A kombinációk száma a két előző számhoz kapcsolódó zárójelezési lehetőségek számával egyenlő.

Az algoritmus működése során mindig az aktuális pozícióban lévő Catalan-számot számoljuk ki az előző Catalan-számok és az előző számok alapján, majd elmentjük azt a táblázatban vagy tömbben. Így minden újabb Catalan-számhoz csak egyszer kell számítást végezni, és a korábban kiszámolt értékekhez könnyen hozzáférhetünk.



5.9. ábra. Catalan számok

Az algoritmus futási ideje lineáris, mivel minden Catalan-szám kiszámításához csak egyszer kell hozzáférni a táblázatban vagy tömbben tárolt előző értékekhez.

Ez a megoldás sokkal optimálisabb mint a rekurziós megoldás, főleg nagyobb számsorozatok esetében.

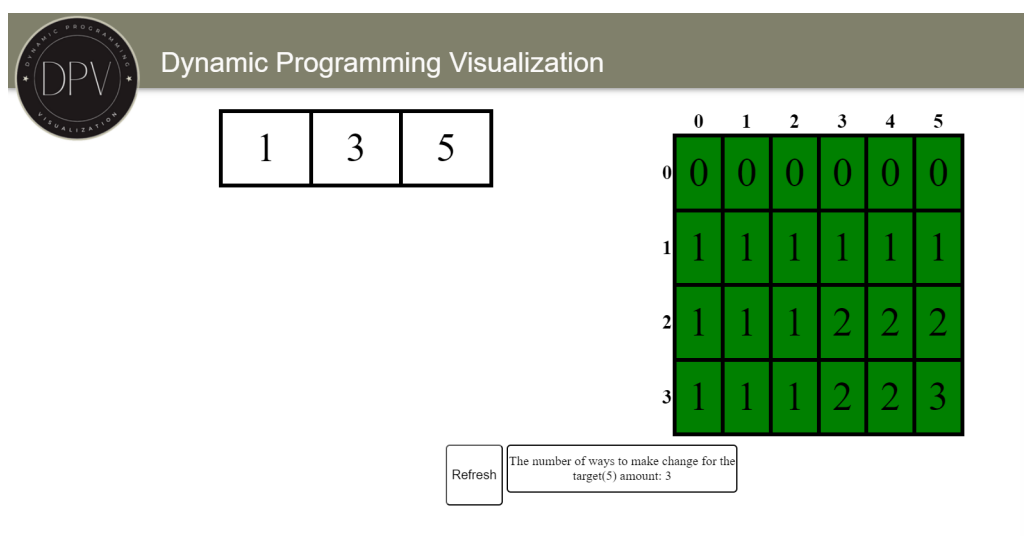
5.3.4. Pénzértékes feladat(Mátrixos megoldás)

A pénzértékes feladat egy olyan probléma, amelyben adottak a rendelkezésre álló érmék névértékei és egy célösszeg. A feladat az, hogy meghatározzuk, hányféleképpen lehet összeállítani a célösszeget az adott érmékből. A dinamikus programozás algoritmusával hatékonyan megoldhatjuk ezt a problémát. Ezt a feladatot meglehetősen oldani egy dimenziós tömb, valamint két dimenziós tömb segítségével is, a dolgozatomban a mátrixos verziót választottam.

Az algoritmus működése során iteratíván töltjük fel a dinamikus táblázatot. Először inicializáljuk az első sort és az első oszlopot. Az első sor minden cellájában 0 szerepel, mivel a 0 összeget nem lehet megszerezni egyetlen érmevel sem. Az első oszlop minden cellájában 1 szerepel, mivel a célösszeg 0 esetén csak az üres halmazt jelenti.

Ezután folytatjuk az iterációt a többi sor és oszlop esetében. Minden cellát az előző sor és az aktuális oszlop értékei alapján számítunk ki. Ha az aktuális célösszeg nagyobb vagy egyenlő az aktuális érme névértékével, akkor a cella értéke a táblázat azonos sorának azonos oszlopában lévő érték és az aktuális célösszeg mínusz az aktuális érme névértéke közötti különbség összege.

Az algoritmus végén a mátrix utolsó sorának utolsó oszlopában található érték lesz a válasz, vagyis az összes lehetséges kombináció száma.



5.10. ábra. Pénzértékes feladat példa

Az algoritmus futása során az eredmények és a számítások lépésről lépésre megjelennek a felhasználói felületen. Az egyes cellák értékei és állapotai (aktuális, keresés stb.) láthatók a táblázatban. A számítások közben egy szövegdobozban látható, hogy éppen melyik résznél tartunk.

A táblázatban tárolt értékek és az egyes lépések vizuális megjelenítése segíti a megértést és az eredmények nyomon követését.

Összefoglaló

Összességében, a dinamikus programozás rendkívül hasznos és sokoldalú eszköz a mindennapi életben. Az algoritmus által kínált optimalizálás és hatékonyság segíthet a feladatok és döntések meghozatalában, idő- és erőforrás-takarékosságot eredményezve. A dinamikus programozás alkalmazása lehetővé teszi számunkra, hogy a komplex problémákat megoldjuk egyszerűbb részekre bontva, és ezáltal sikeresebben navigáljunk az élet különböző területein.

A dolgozatom során a dinamikus programozás vizualizációs weboldal elkészítésével foglalkoztam, amelynek célja az volt, hogy interaktív módon mutassa be a dinamikus programozás algoritmusait és azok működését. A fő feladatom az volt, hogy készítsek egy olyan webalkalmazást, amely segítségével a felhasználó könnyen megértheti és követheti az algoritmusok működését. Első lépésben elkészítettem a weboldal felületét, amely magában foglalta a logót, a fejléct és az opció menüt a fent említett dinamikus programozás problémákhoz. A logó és a fejléc segített a weboldal egyedi megjelenésében és az azonosításban, míg az opció menü lehetővé tette a felhasználó számára, hogy válasszon a különböző feladatok között. Ezután nekiláttam a különféle dinamikus programozás által megoldott feladatok implementálásának és bemutatásának. A minimum/maximum költségű utak meghatározásához két dimenziójú tömböt alkalmaztam, ahol vizualizáltam az értékek kiszámításának módját, meghatároztam a konkrét utat, valamint a végső költséget is kimutattam. A Fibonacci sorozat megoldásához egy egy dimenziós tömböt használtam, ahol minden cellában az adott Fibonacci szám volt látható. A Catalan számok megoldása hasonló elveken alapult. Az alkalmazás lépésről lépésre bemutatta a Catalan számok kiszámítását. A felhasználó megadhatott egy számot, és a vizualizálás során megfigyelhette, hogyan számolodnak ki a Catalan számok. A pénzürmés feladatban azt kellett meghatározni, hányféleképpen lehet összeállítani egy adott célösszeget adott érmék használatával. Ehhez egy mátrixot használtam, ahol minden cella az adott célösszeghez tartozó értéket tartalmazta. A táblázat és egy szövegdoboz segítségével megjelenítettem a lépéseket és az eredményeket. Az említett feladatok implementálásához a React keretrendszert használtam, amely lehetővé tette a dinamikus és interaktív felhasználói felület létrehozását. A HTML, CSS, Typescript és MUI komponensek segítségével erős típusrendszert és kiterjesztett nyelvi lehetőségeket biztosítottam a fejlesztés során, valamint felhasználó barát és edukatív jellegű weboldalt hoztam létre.

Ami a jövőbeli terveimet illeti a projektemmel kapcsolatban, mindenképp szeretném tovább bővíteni a feladatok katalógusát izgalmasabb vizualizációkkal, valamint fejleszteni a weboldal interaktivitását. Továbbá fontosnak gondolom, hogy a weboldalon bemutatott feladatok folyamatosan fejlődjenek, hogy minél érthetőbbek legyenek főként a diákoknak és a dinamikus programozást érdeklők körében.

GitHub repository **link:** <https://github.com/StoicaAttila/Dynamic-Programming-Visualization.git>

Köszönetnyilvánítás

Szeretném megköszönni a projektvezető tanáromnak, Dr. Kátai Zoltánnak a hasznos tanácsait, támogatását és biztatását, mellyel magasabb lécre léphettem. Köszönöm, hogy bármikor fordulhattam hozzá ha kérdések merültek fel és mindig nyitott volt meghallgatni és elmagyarázni amíg ki nem tisztázodtak a felmerült témák.

Hálás vagyok a szüleimnek, a barátaimnak és legfőképpen a barátnőmnek a határtalan kitartásukért, türelmükért és segítségükért amiért mellettem álltak jóban rosszban és végtelen erővel és szeretettel töltötték fel. Átaluk az utamon levő akadályokat sokkal bátrabban és határozottabban tudtam felülmúlni.

Ábrák jegyzéke

3.1. Use Case	16
3.2. Architektúra diagram	17
3.3. Szekvencia diagram	18
4.1. Projekt html struktúrája	20
4.2. Projekt struktúrája	22
5.1. Dynamic Programming Visualization logo	25
5.2. Fejléc	26
5.3. Opció menü	26
5.4. Page	27
5.5. Refresh gomb és művelet magyarázó	28
5.6. Minimum költségű út példa	29
5.7. Maximum költségű út példa	29
5.8. Fibonacci számok	31
5.9. Catalan számok	32
5.10. Pénzérmés feladat példa	33

Irodalomjegyzék

- [1] Kátai Zoltán. *"Algoritmusok felülnézetből"*. 2007. *Scientia Kiadó*.
- [2] Meenakshi and Kamal Rawat. *"Dynamic programming for coding interviews, a bottom-up approach to problem solving"*. 2017. *Notion Press*.
- [3] Dimitri P. Bertsekas. *"Dynamic Programming and Optimal Control"*. 2017. *Athena Scientific*. Vol. I. *4th*.
- [4] Anany Levitin. *"Introduction to the Design and Analysis of Algorithms"*. 2002. *Addison Wesley*. *United States Ed*.
- [5] Robert Sedgewick and Kevin Wayne. *"Algorithms"*. 2011. *Addison-Wesley Professional*. *4th*.