AlgoRythmics: Tánctól a Kódig

Osztián Pálma Rozália[1], Kátai Zoltán[2]

{[1]osztian.palma, [2]katai_zoltan}@ms.sapientia.ro
SAPIENTIA EMTE

Absztrakt. Köztudott, hogy napjainkban jelentősen elterjedt az online oktatási környezetek használata, és bár folyamatosan bővül az E-learning környezetek tárháza a kérdés mindig megfogalmazódik bennünk: Vajon hogyan lehetne a lehető leghatékonyabban tanítani informatikát és fejleszteni az algoritmikus-gondolkodást?

Az elmúlt időszakban kifejlesztettünk egy olyan online oktatási környezetet, mely hasonló a közismert E-learning felületekhez, mégis különleges a maga módján. *AlgoRythmics*: az *algoritmus* (informatika) és a *ritmika* (ritmus, tánc), vagyis a *tudomány* és *művészet* ötvözete. Egyedisége abban rejlik, hogy különböző algoritmusok és azokhoz tartozó tanfolyamok segítségével segít a felhasználóknak az algoritmikus gondolkodás fejlesztésében úgy, hogy elvezeti őket a *tánctól, egészen a kódig*. Mindezt öt alapvető tanulási lépés teszi lehetővé: *a videó, animáció, levezénylés, kódépítés és az életre kelt kód* fázisai.

Az egyedi tanulási lépéseknek köszönhetően a teljes tanulási folyamatot változó *interaktivitási szint* jellemzi, így a tanfolyamok során a felhasználóknak lehetőségük van független megfigyelőként (*0 interaktivitás*), részleges felhasználói irányítással (*1/2 interaktivitás*), illetve teljes felhasználói irányítással (*1 interaktivitás*) végig vezetni az algoritmus lépéseit.

Kulcsszavak: E-learning, interaktivitás, algoritmikus-gondolkodás, tanulási lépések, tudomány és művészet

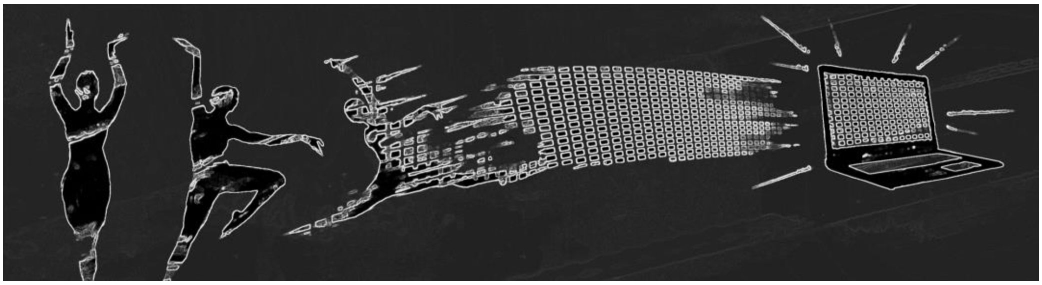
1. Bevezetés

Mai digitális világunkban egyre többen, akár már gyerekkorban, találkoznak a számítógépek és az informatika nyújtotta lehetőségekkel. Nagyon sok gyerek már egészen kiskortól telefont fog a kezében, vagy számítógép elé ül, és játszik, tanul, fejlődik mely cselekvésekhez elengedhetetlen, hogy olykor használja, talán tudattalanul is, számítógépes gondolkodását. Amint azt Jeannette Wing is megfogalmazta *a számítógépes gondolkodás* a *negyedik* alapvető készség, az *aritmetika*, *olvasás* és az *írás* mellett, mellyel egy XXI. századi embernek rendelkeznie kell, és folyamatosan fejlesztenie kell azt. Ennek a készségnek a fejlesztésére az egyik legnagyobb segítséget az informatikai algoritmusok nyújtják, melyek nap, mint nap, jelentős részét képezik életünknek. Elsősorban ezek megértése feltételezi a tudásban való elmélyülést. Emellett azonban sok szakértő úgy tekinti, hogy a számítógépes gondolkodás hatékony fejlesztése magába foglalja a kódolást is.

Napjainkban egyre több iskolában vezették be a kisiskolások órarendjébe is az informatika órákat. Erdélyben már a 2017/2018 tanévtől kezdődően jelen van az informatika óra a középiskolások tantervében. Belátható, hogy az informatikai algoritmusok megértése és elsajátítása fontos szereppel bír. Ebben a szellemben vette kezdetét 2007-ben az *AlgoRythmics* projekt, mely kapcsán 6 videó és egy webalkalmazás látott napvilágot. A videók 6 *rendezési algoritmus* (buborékos-, beszúró-, kiválasztó-, összefésülő-, gyors- és shell rendezés) néptánccal eltáncolt változatát mutatták be. Amint azt a neve is mutatja ez az oktatási stratégia a **tudományt** *(algoritmus)* és a **művészetet** *(tánc, vagyis ritmus)* ötvözte, így kapta az *Algo – Ritmika*, vagy angolul *AlgoRythmics* nevet.

Az előzetes kutatásoknak és munkának köszönhetően az algoritmusokat bemutató tánc-videók rendelkezésünkre álltak, és amint az kiderült hasznosak is bizonyultak. Szerettük volna azonban ezeket egybegyűjteni egy olyan oktatási környezetben, mely más módszerekkel kiegészülve, kéz a kézben, segítséget nyújt a felhasználóknak az algoritmikus gondolkodás elsajátításában. Egy olyan E-learning környezet implementálását tűztük ki célul, mely hasonló, de mégis a maga nemében más tanulási lépések révén vezeti el a felhasználókat a tánctól, egészen a kódig úgy, hogy közben az interaktivitás különböző szintjein nyilvánul meg.

Bár algoritmus-vizualizációval számos helyen találkozhattunk már, ez a környezet a táncra és az animációkra összpontosul. Ezek közös bemutatásával akár két eltérő tanulási stílust is megcélozhatunk, hiszen lehetőséget biztosítunk a felhasználóknak arra, hogy *dinamikus (tánc)*, illetve *statikus (animáció)* vizualizációval tanulhassanak. A dinamikus bemutatás magával hozza az emberi mozgás effektust, mely segítségével a diákok azonosulni tudnak az adott számot viselő táncossal, a statikus vizualizáció pedig egy absztraktabb bemutatást tesz láthatóvá, ahol az animáció különböző, letisztult és egyszerű mozzanatai érvényesülnek.



1. ábra: Tánctól a Kódig

2. Interaktivitási szintek

0 Interaktivitás:

Ez a törzsfogalom foglalta magába a független megfigyelést, mely során a felhasználók, vagyis a diákok zavartalanul tekinthették meg egy-egy algoritmus bemutatását (videó vagy animáció által). Jelen oktatási rendszerünkben ez az előadó órákhoz hasonlítható, melyek esetében a tanár előadást tart egy adott témáról, a tanulók pedig figyelemmel kísérik azt.

½ Interaktivitás:

Az úgynevezett „fél" interaktivitás, a felhasználók részleges bevonását jelentette. Jelen napjainkban mindez párhuzamba hozható az interaktív tanórákkal, mely során a tanárok kérdéseket intéznek a diákokhoz, ezáltal próbálva bevonni őket az tanítási- és tanulási folyamatba.

1 Interaktivitás:

A harmadik és egyben utolsó csoportosítási osztály a teljes felhasználói irányítást foglalta magába. Ez tulajdonképpen a teljes interaktivitást jelentette, mely során a tanulóknak önállóan kellett *„levezényelni"*, vagyis irányítani az adott algoritmust. Mindennapjainkban ez önálló munkák, feladatok formájában jelenik meg, mely során a diákoknak a kitűzött feladat kezdetétől el kell jutniuk a végső termékig, vagyis az elkészített feladványig.

3. Szakirodalmi áttekintő

3.1. 0 Interaktivitás

Számos tanulmány bizonyította, hogy az algoritmusok animációval történő reprezentációja igen hatékony és elősegíti a hallgatókat abban, hogy megértsék az algoritmusok működését. A kutatók azt is megfigyelték, hogy egyszerű animációk által nem érhetünk el tartós tudást, hiszen a legtöbb esetben a hallgatók néhány hónap után már el is felejtették az algoritmust. Annak érdekében, hogy az algoritmusokkal kapcsolatos tartós ismeretek megszerzését biztosítsunk, a nézőket be kell vonni jelentőségteljes interakciókkal is. Bebizonyosodott, hogy azon hallgatók emlékezetében, akiknek van lehetőségük megtapasztalni, *„átélni”* egy adott témával kapcsolatos tudnivalókat, sokkal hosszabb ideig megmarad az információ, mint azokéban, akikkel csak az elméleti aspektusokat szemléltették, jelentőségteljes interakciók nélkül.

3.2. ½ Interaktivitás

Egy módszer, amely segítségével be lehet vonni a diákokat a tanulási folyamatba, az úgynevezett „interaktív jóslás”, mely azt jelenti, hogy az animációs folyamat megszakad, és a felhasználónak kell meghatároznia a következő mozzanatot [1]. Ezzel kapcsolatosan számos kutatást vizsgáltunk meg: Először Byrne, Catrambone és Stasko [2] dolgozott ki egy módszert, melynek segítségével bevonták a hallgatókat a megjelenítési folyamatba. Az előre meghatározott pillanatokban a hallgatóknak szóban kellett elmondaniuk, hogy mi fog történni. A kutatás eredményei azt mutatják, hogy a hallgatóknak sokáig megmaradt emlékezetükben mindaz, amit megtanultak. Korhonen és Malmi kutatása [3] egy olyan módszert mutat be, mely során a tanulóknak lehetőségük nyílt az algoritmus manuális levezénylésére. Adott kulcsmomentumokban válaszolniuk kellett kérdésekre, melyeket rögzítettek és azonnali visszajelzést kaptak rá diákok. A kutatás nagyon jó eredményekhez vezetett. Egy újabb ötletes módszernek számít Naps, Eagan és Norton [4] kutatása, akik az algoritmus vizualizáció során „stop-and-think” kérdéseket használtak. Ahhoz, hogy a tanulók folytatni tudják a tanulási folyamatot, válaszolniuk kellett a feltett kérdésekre. Ezesetben is automatikusan kiértékelődtek a válaszok és visszajelzést kaptak a diákok, a tanulási folyamat pedig igen hatékonynak bizonyult.

Ezek mellett, Jarc, Feldman és Heller tanulmányában bemutatott megjelenítési rendszer [5] szintén megpróbálta aktívan bevonni a hallgatókat jelentőségtejes interakciókkal. Rendszerük lehetővé tette a hallgatók számára, hogy passzív módon tekintsék meg a vizualizációt *(„Mutasd meg”)*, és úgy is, hogy legyenek bevonva interaktív kérdésekkel *(„Megpróbálom”)*. Meglepő módon kutatási eredményeik azt mutatják, hogy a nagyobb interaktivitású csoport következetesen, de nem szignifikánsan alulteljesítette a kontroll csoportot.

3.3. 1 Interaktivitás

Továbbá, egy újabb tanulmány is megerősíti az interaktív algoritmusok interaktív megjelenítésének hatékonyságát. Ennek kapcsán nyer először értelmet az *„orchestration”* algoritmus fogalma, mint magasabb szintű interaktivitás: *„Az interaktív predikció egyik különleges esete az, amikor a hallgatóknak a vizsgált algoritmust kell irányítaniuk”*. Ez tulajdonképpen azt jelenti, hogy az interaktív vizuális tanulási környezetet használó hallgatóknak, az algoritmusnak nem csak a következő lépését kell meghatározniuk, hanem végre is kell hajtaniuk azt. A feladat megértésének és teljesítésének biztosítása érdekében a környezetnek azonnali visszajelzést kell adnia minden beérkezett válaszról, és lehetőséget kell, hogy biztosítson arra, hogy a tanulók újra próbálkozhassanak, és szükség esetén segítséget kérhessenek. A kutatás egyik fő gondolata kihangsúlyozta, hogy a *„levezénylési”* folyamat elsődleges célja nem a felmérés, hanem a hallgatók megértésének javítása és finomítása a vizsgált algoritmusról [1].

Mivel megoszlanak a vélemények az oktatásban használt interaktivitás mértékéről, a fent említett tanulmányok alapján azt mondhatjuk, hogy minden egyes interaktivitási szintnek megvannak a saját erősségei és gyengeségei, és nincs univerzálisan optimális interaktivitási szint. Éppen ezért döntöttük úgy, hogy ennek hasznosságát és szerepét szeretnénk az általunk implementált AlgoRythmics oktatási környezetben is tesztel, hiszen annak egyik jellegzetessége a változó interaktivitási szint.

4. AlgoRythmics online oktatási környezet

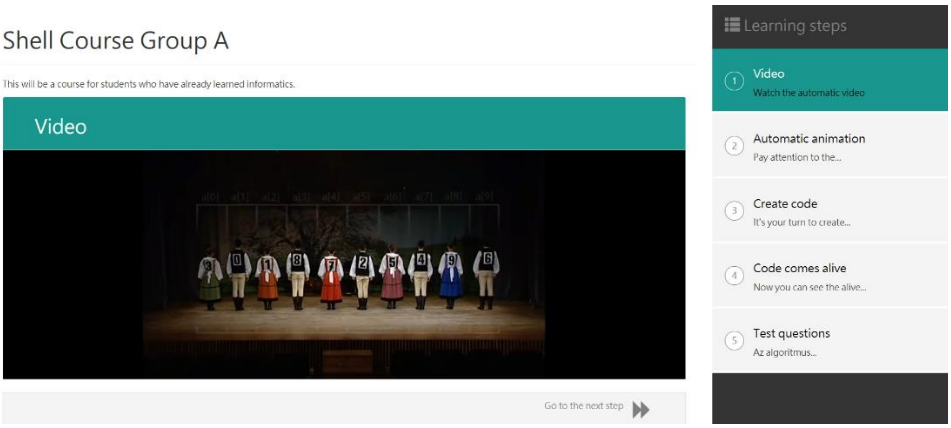
Az AlgoRythmics oktatási környezet hasonló más online, tanítást és tanulást elősegítő oldalakhoz, mégis különlegesnek nevezhető a maga módján. Elsősorban a rendezési-, keresési- és visszalépéses keresési algoritmusokra specializálódott. Célunk az volt, hogy egy olyan interaktív weboldalt biztosítsunk, mely elősegíti a tanulók algoritmusokban való elmélyülését, azok megértését és végül, de nem utolsó sorban leprogramozását. Mindezt, az oktatási környezetet alkotó algoritmusok, tanfolyamok és a tanulási lépések segítségével valósítja meg.

4.1. Tanulási lépések

Az egyik legfontosabb összetevői az oldalnak a tanulási lépések. Donald Knuth *„Egy algoritmust látnom kell ahhoz, hogy elhiggyem."* gondolatára alapozva, kijelenthető, hogy az algoritmusok vizualizációja nagymértékben hozzájárul a megértéshez, de emellett a tanulók kíváncsiságának, motivációjának felkeltéséhez is. Az *5+1 tanulási lépés* tulajdonképpen ezt a célt, valamint a lecketervek változatosságát biztosítja, hiszen ezek mindegyike más és más jellegzetességeivel járul hozzá a tanulási folyamat fázisaihoz.

4.1.1. Videó

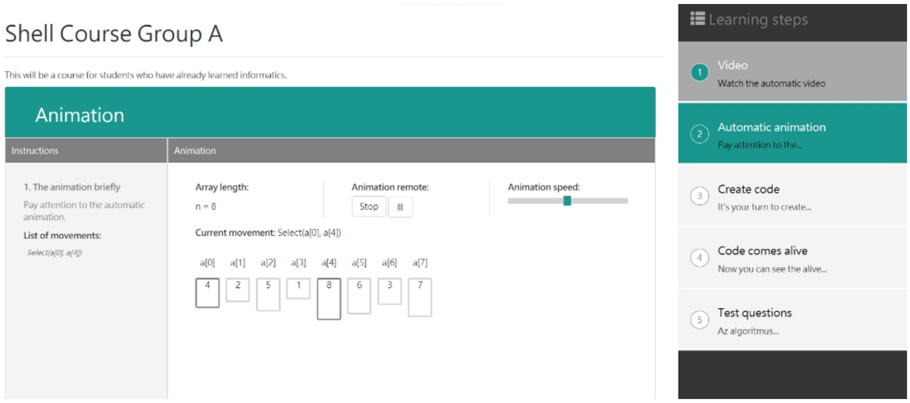
A videó tanulási lépés mutatja be az AlgoRythmics kutatócsoport által létrehozott eltáncolt algoritmusok videóit. A videók amellett, hogy egy-egy algoritmus vizualizációját jelentik, etnikai jelleggel is rendelkeznek. A különböző táncstílusok, melyek megtalálhatóak a táncvideókban, illeszkednek az adott algoritmusok típusaihoz. A rendező algoritmusok erdélyi néptánccal (pl: küküllőmenti-, szászcsávási-, mezőségi néptáncok), a kereső algoritmusok flamenco tánccal, míg a visszalépéses kereső algoritmusok (N királynő probléma) balett tánccal vannak bemutatva.



2. ábra: Videó tanulási lépés

4.1.2. Animáció

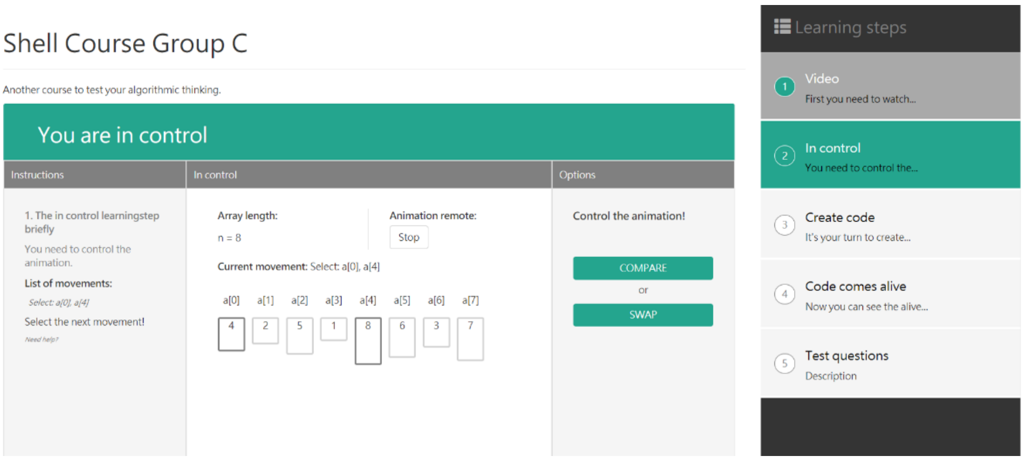
Az animációk képezik az algoritmusok bemutatásának egy absztraktabb változatát. A korábban emberek által képviselt értékek helyét, az animáció esetén számértékek veszik fel, melyek különböző magassággal rendelkező „dobozok” segítségével utalnak az adott értékek nagyságrendjére. Ez a tanulási lépés az egyik legfontosabb részét képezi az oktatási környezetnek, hiszen a videó tanulási lépésen kívül minden másik tanulási fázisban megjelenik, és ehhez illeszkednek, illetve társulnak az algoritmusvizualizáció további jellemzői.



3. ábra: Animáció tanulási lépés

4.1.3. Levezénylés

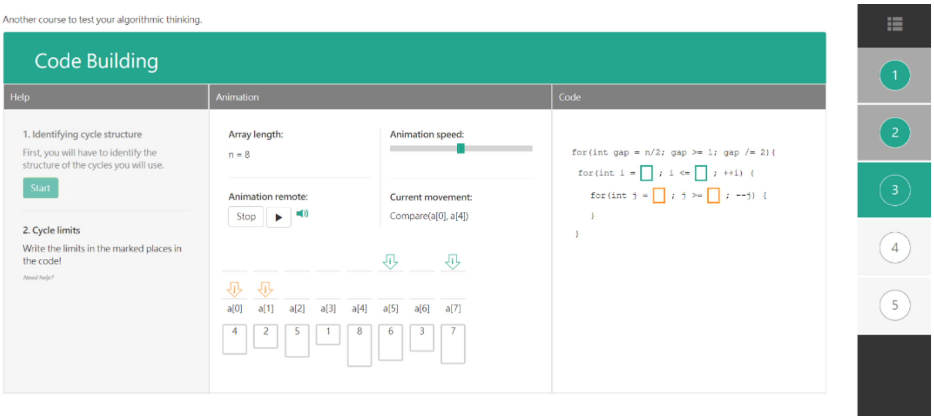
A levezénylés tanulási lépés az animáció egy komplexebb változata. Az előző tanulási lépéshez hasonlóan, itt is megjelenik egy „dobozok” által szemléltetett számsorozat, melyet rendezni, vagy melyben keresni kell. Ezúttal azonban teljes szinttű interaktivitás jellemzi a tanulási folyamatot. A felhasználónak elejétől a végéig kell irányítania az algoritmus főbb mozzanatait. Mindezt, az alapvető műveleteket segítő gombok használatával, vagyis az összehasonlítás (compare), csere (swap) gombokkal, illetve az elemek kiválasztásával teheti meg.



4. ábra: Levezénylés tanulási lépés

4.1.4. Kódépítés

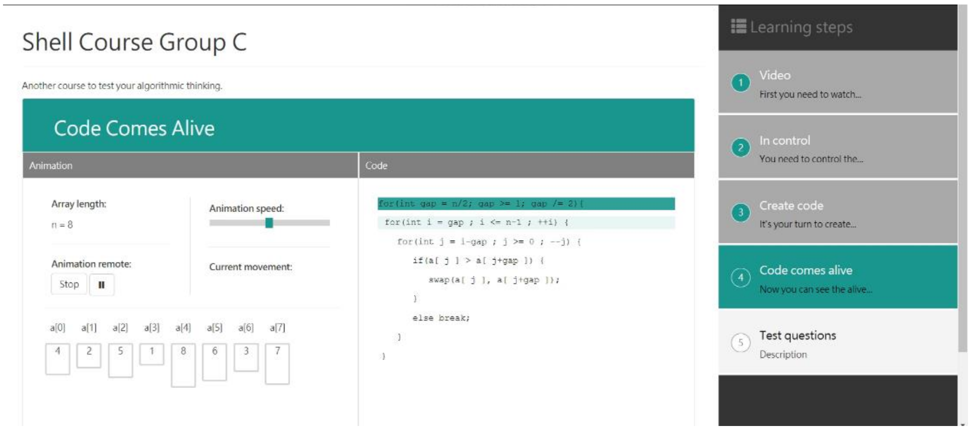
Az első három tanulási lépés segíti elő az algoritmikus-gondolkodásban való elmélyülést, a szemléltetést és ez által az algoritmusok megértését. Ahhoz azonban, hogy a tanulók programozási készségeit is tudjuk fejleszteni a bemutatott algoritmusokat illetően, bevezettük a kódépítés fázisát is. Bár már a videó tanulási lépés során is megjelenik a *hang*, a *zene*, itt a hallással való érzékelés külön szerepet kap. Ez az adott algoritmus ciklusszerkezetének meghatározásakor lép érvénybe, amikor a felhasználónak hang alapján kell meghatározni, hogy egy, két egymásba ágyazó, vagy két külön ciklus képezi az algoritmus vázát. Ezt követően az adott ciklusok határértékeit kell, hogy meghatározza a felhasználó, majd az összehasonlításra, illetve a cserére vonatkozó kritériumokat. Ha mindezt sikeresen elvégezte, elkészül az algoritmushoz tartozó kódrészlet.



5. ábra: Kódépítés tanulási lépés

4.1.5. Életre kelt kód

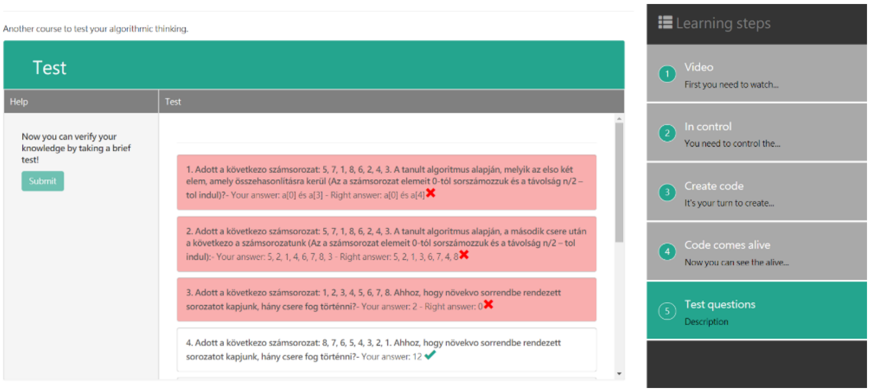
Ez a tanulási lépés tekinthető egyfajta összefoglalónak is. Az animáció, itt is fontos szereppel bír, hiszen a „megépített" kód egy-egy mozzanata egyszerre értékelődik ki az animáció különböző fázisaival. Tulajdonképpen olyan, mintha a kód irányítaná az animációt.



6. ábra: Életre kelt kód tanulási lépés

4.1.6. Kiértékelő tesztek

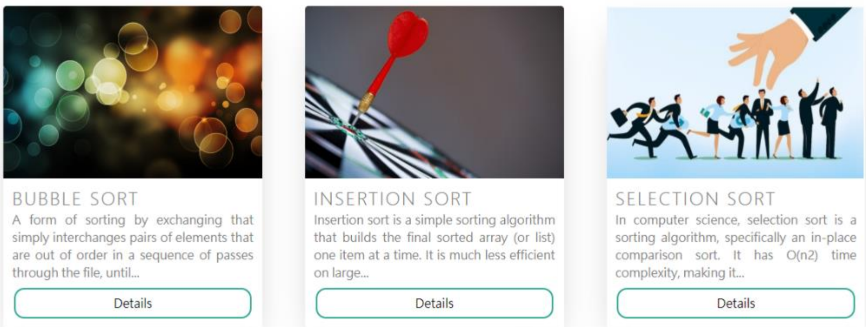
Az öt alapvető tanulási lépés segít a felhasználóknak abban, hogy elmélyüljenek az algoritmusok különböző vizualizációjában. Ahhoz azonban, hogy fel tudjuk mérni a tanulók teljesítményét is, azt hogy mennyire értették meg az adott algoritmus lényegét, és a különböző interaktivitási szintek mindegyike milyen mértékben bizonyult hatékonynak, bevezettük az ellenőrző tesztek fogalmát, vagyis egy extra (+1) tanulási lépést. Ezeket tetszés szerint határozhatják meg a tanárok különböző válaszlehetőség típusokat megadva (egy-, több- vagy véleménynyilvánító válaszok). A válasz-lehetőségek meghatározása mellett, a tanárok a kérdések számát is rögzíthetik. A felhasználók, amennyiben minden kérdést megválaszoltak, megtekinthetik helyes, illetve helytelen válaszaikat, hiszen a felmérő tesztek automatikusan értékelődnek ki.



7. ábra: Kiértékelő teszt

4.2. Tanfolyamok

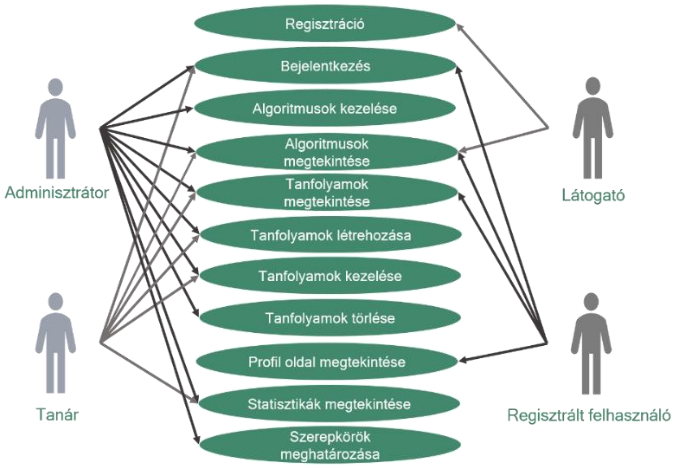
Az oldalon megjelenő online tanfolyamok teszik lehetővé azt, hogy mérni tudjuk a felhasználók teljesítményét. A tanfolyamok tulajdonképpen a felsorolt tanulási lépések különböző változataiból és kombinációiból épülnek fel, melyeket a tanárok vagy adminisztrátorok állíthatnak be tetszés szerint. Ehhez kapcsolódik az interaktivitási szintek meghatározása is. A tanulási lépések és azok típusai megfelelnek egy-egy adott interaktivitási szintnek, így azok hozzáadása a tanfolyamokhoz egy meghatározott interaktivitással rendelkező lecketervet eredményez.



8. ábra: Tanfolyamok

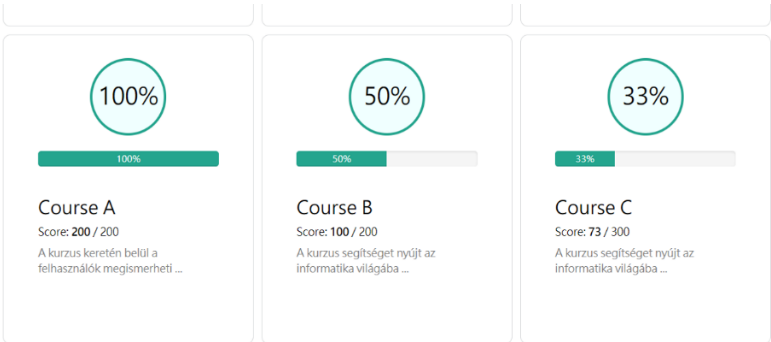
4.3. Felhasználók

Jelen oktatási rendszerünk két legfontosabb szereplője: a diák és a tanár. Ezt a két fő kategóriát az általunk implementált E-learning környezetben kibővítettük, és így négy felhasználói szerepkört különítettünk el. Az *adminisztrátor* és a *tanár* foglalja magába a **„szerkesztő"** funkcionalitásokat, melyek a lecketervek és azok jellemzőit határozzák meg; a *látogató,* illetve a *regisztrált felhasználó* (az adott tanuló) pedig a **„megfigyelő"** lehetőségeket, vagyis megtekinthették, és elvégezhették a rendelkezésükre álló tanfolyamokat.



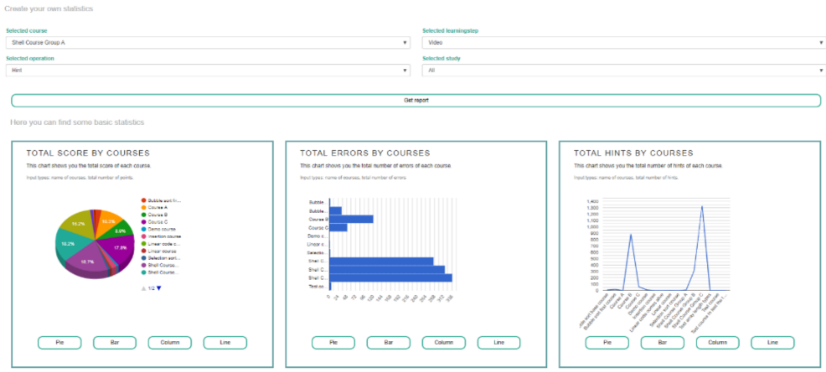
9. ábra: Use case diagram

Tanulási folyamatról lévén szó, úgy a tanárok, mint a tanulók is kíváncsiak elért eredményeikre. A diákok számára biztosítottunk egy *felhasználói* (profil) *oldalt*, melyen minden regisztrált felhasználó nyomon követheti az elkezdett tanfolyamokon való előrehaladást, és az elért pontszámot.



10. ábra: Profil oldal

A tanulási folyamat eredményességét a hibák, illetve segítségkérések összességével határoztuk meg. Minden felhasználói interakció során kérhettek segítséget a tanulók az adott lépést illetően, illetve hibázhattak. Ezek közül mindkettő pontlevonással járt. Annak érdekében, hogy a tanárok is megtekinthessék az adott lecketervek hatékonyságát, és azt, hogy az algoritmusok mely mozzanatai okozták a legnagyobb nehézséget a diákoknak (milyen gyakori volt a segítségkérések és hibák száma) lehetőséget biztosítottunk statisztikák megtekintésére, melyek az addigi felhasználói interakciók alapján értékelődnek ki.

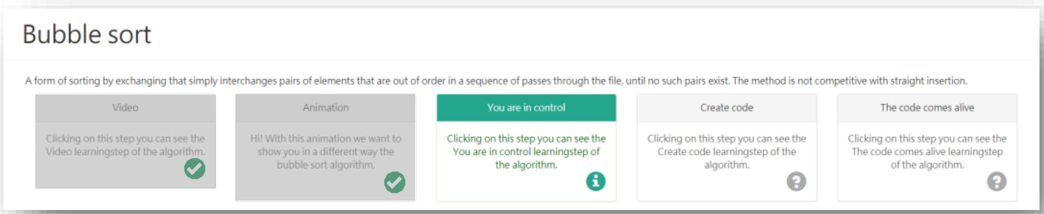


11. ábra: Statisztikák

4.4. Megjelenítési követelmények

4.4.1. Algoritmusok

Az algoritmusok esetében elsősorban az elméleti tudást szerettük volna megalapozni, így az öt tanulási lépés bemutatása előtt a felhasználók megtekinthetik az algoritmus nevét és a hozzá tartozó rövid leírást. Ez egyben, egy rövid ismerkedést is jelent az adott algoritmussal kapcsolatosan. A leírást a tanulási lépések öt típusa követi, melyek egy-egy kapcsolótáblával vannak szemléltetve. Bár a tanulási lépések elvégzésének sorrendje nincs meghatározva, a rendszer nyomon követi a felhasználók cselekvéseit. A még meg nem tekintett tanulási lépések kapcsolótábláit világosszürke szín és egy kérdőjel, az aktuális tanulási lépést zöld szín és egy információs ikon, míg a már megtekintett tanulási lépéseket sötétszürke szín és egy pipa jelölte.



12. ábra: Tanulási lépés kapcsolótábláinak megjelenítése

A tanulási lépések kapcsolótábláit követően az aktuális kiválasztott lépés jelent meg. Ez az animáció esetén két részre (feladatok és animálandó számsorozat), levezénylés esetén pedig három részre (feladatok, animálandó számsorozat és lehetséges műveletek gombjai) oszlott.

Az oldal alját az algoritmushoz tartozó speciális tanfolyamok listája zárta. Azok megtekintéséhez szükséges volt az oldalra látogató bejelentkezése.

4.4.2. Tanfolyamok

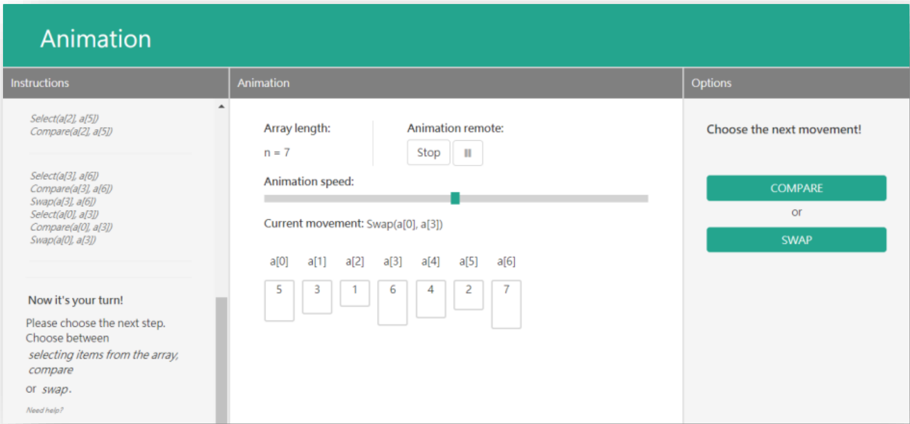
A tanfolyamok esetén az algoritmusokkal ellentétben nem egymás alatt követte a tanulási lépések listája és az aktuális lépés részletes bemutatása egymást, hanem egymás mellett. Ez azért volt fontos, mert a felhasználóknak egyaránt szemléltetni szerettük volna az adott tanulási lépés fázisát, de azt is, hogy hol tart a tanfolyam elvégzésével kapcsolatban.

Mivel ennek következtében kevesebb hely jutott a soron következő tanulási lépés részletezésére, azt is megvalósítottuk, hogy a lépések listája „becsukható" (elrejthető) legyen. Így sikerült több helyet biztosítani az animációknak és lehetséges műveleteket jelző gomboknak.

4.4.3. Tanulási lépések

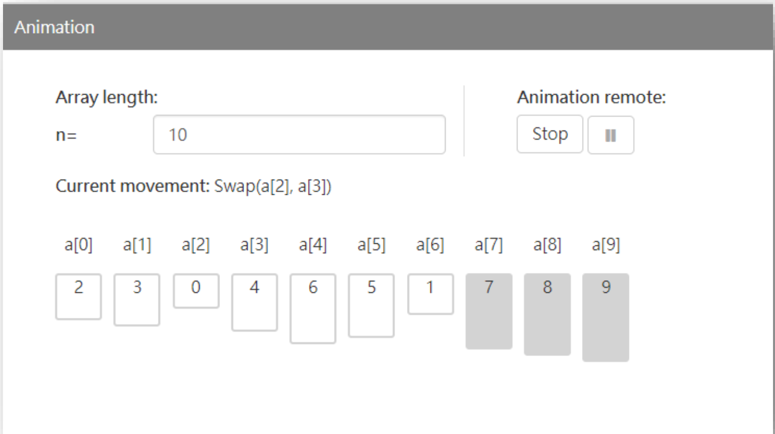
Szinte minden tanulási lépésre meg tudtunk határozni egy általános megjelenítési jellemzőt. A video kivételével minden lépés esetében legalább két részre osztottuk a képernyőt, melyek közül az első rész mutatta be a szükséges feladatok leírását, melyet a felhasználónak végig kellett vezetnie. Emellett, itt volt lehetősége a felhasználónak segítséget kérni a következő helyes műveletet illetően. Ez a rész biztosította azt is, hogy a tanuló nyomon követhesse az adott lépés fázisait (az eddig megtörtént mozzanatokat és műveleteket), és azt, hogy mikor bizonyul teljesítettnek az adott tanulási lépés.

A második rész legfontosabb részét az animálandó számsorozat képezte. Ez dobozok és azokban szereplő értékek segítségével volt szemléltetve. Az animáció fölött közvetlenül megjelenítettük a számsorozatot képviselő tömb jelét (a) és minden elem fölött az aktuális pozíciót (pl.: a[0], a[1], a[2]…). A második rész fontos jellemzőit képezte a számsorozat hosszának, illetve az aktuális művelet nevének megjelenítése. A felhasználóknak lehetőséget adtunk az animáció sebességének változtatására is, melyet egy úgynevezett sebességsáv jelezte. Kezdetben az animáció sebességét közepesre állítottuk. Emellett a felhasználó megállíthatta és újra is kezdhette az animáció futását a **STOP**, **START**, **PAUSE** és **PLAY** gombok segítségével.



13. ábra: Animáció megjelenítésének jellemzői

Rendezésekről lévén szó, fontosnak találtam valamiképp jelezni azt, ha egy sorozat béli elem helyére került. Ezt sötétebb szürke háttérrel rendelkező dobozzal jelöltem. Így a felhasználó számára világossá vált az, hogy az sötétebb szürke elemeket már nem szükséges tovább hasonlítani vagy cserélni, hiszen a helyükre kerültek.



14. ábra: Helyes pozíción található elemek szemléltetése

Az adott lépések interaktivitási szintjének függvényében a megjelenítések típusa is változott. Ezek kapcsán a következőket kellett biztosítanunk:

Videó esetén:

- A felhasználónak látnia kell a videót teljes méretében, és ha szeretné, akkor kinagyított formában is.
- Interaktív videó esetén a kérdéseknek nem szabad eltakarniuk az épp zajlódó mozzanatokat annak érdekében, hogy a felhasználó a lehető legjobb választ tudja megadni.

Interaktív animáció és levezénylés esetén:

- Mindkét tanítási lépés esetében kell, hogy legyen egy harmadik rész, amely az irányításért felel. Ebben a részben találhatja meg a felhasználó az összes olyan utasítást, melyre szüksége van az adott lépés véghezviteléhez (összehasonlítás és csere műveletét). Az elemek kiválasztását a középső részben található animálandó számsorozatra kattintva hajthatta végre.

Kódépítés esetén:

- A kódépítés tanulási lépés az előző interaktív lépésekhez hasonlóan három részre osztotta az algoritmus bemutatását: feladatok megfogalmazása, animáció és a kiegészítendő kód. Ennek kapcsán külön szerepet kaptak a színek, melyeket az adott paramétereket jelző nyilak és keretek segítségével szemléltettünk. Az i és j mutatók az animáció fölött jelentek meg nyilak formájában. Az ezzel összhangban megjelenő ciklushatárok i és j mutatók színe függvényében váltakoztak. Az i-vel kapcsolatos kiegészítendő rész i pointer színével, a j-vel kapcsolatos információ pedig j pointer színével egyezett meg.
- A ciklusszerkezet meghatározását szolgáló kis ablak az említett három rész (feladatok, animáció és kódrészlet) előtt jelent meg.

Az életre kelt kód esetén:

- Az kód életre keltésének lépésénél az eddig megjelenített feladatok rész eltűnik, hiszen itt a felhasználónak csupán figyelnie kell a kód által irányított animációt. Ennek következtében a segítségkérés és a hibák lehetősége is elmarad, hiszen teljes szinttű automatikus lejátszás jellemző a tanulási lépésre.
- A színek jelen esetben is fontos szerepet kapnak, hiszen az animáció adott fázisai a kód egy bizonyos részével együtt értékelődnek ki, melyet a kódsor kiszínezett keretével szemléltettünk.

4.4.4. Tanulási lépések jellemzői

A tanulási lépések függvényében az animáció tulajdonságainak megjelenítése is változott.

„Fehér mód"

- Az animálandó sorozat elemeit képező dobozoknak a mérete nyilvánvaló kell, hogy legyen a felhasználó előtt, illetve az is, hogy milyen számot tartalmaznak.

„Fekete mód"

- El kell rejteni a felhasználó elől a sorozat elemeinek méretét és a bennük lévő számokat. Így, csak akkor tudja meghatározni a felhasználó a következő helyes műveletet, ha figyelmesen követi végig az összehasonlítások és cserék műveleteit.

Tanári bemenet

- A tanfolyam létrehozásakor a tanár előre meghatároz egy számsorozatot. Ezt megmutatjuk a felhasználóknak, viszont nem adunk lehetőséget arra, hogy megváltoztassák (mint ahogy azt más esetekben megteheti). A számsorozat hosszát jelölő bemenet ez esetben nem szerkeszthető.

Véletlenszerű bemenet

- Az animálandó számsorozat ebben az esetben nem állandó. A felhasználónak meg kell engedni a számsorozat hosszának változtatását, és a „START" gomb megnyomás általi új számsorozat generálást.

Legjobb bemenet

- Előre meghatározott számú elem jelenik meg az animálandó sorozat részeként növekvő sorrendben. A felhasználónak lehetőséget kell adnunk a számsorozat hosszának megváltoztatására, viszont ennek jellege nem változhat: rendezések esetén a számsorozat elemeinek értékei mindig növekvő sorrendben maradnak; lineáris keresés esetén, mindig a keresett elem marad az első helyen.

Legrosszabb bemenet

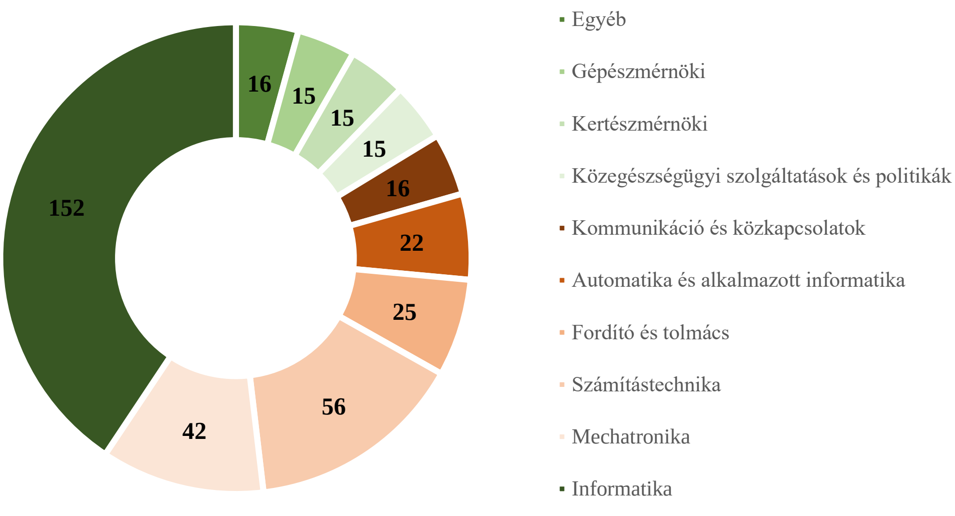
- Előre meghatározott számú elem jelenik meg az animálandó sorozat részeként csökkenő sorrendben. A felhasználónak lehetőséget kell adnunk a számsorozat hosszának megváltoztatására, viszont ennek jellege nem változhat: rendezések esetén mindig csökkenő sorrendben maradnak a számsorozat értékei; lineáris keresés esetén mindig legutolsó helyen szerepel a keresett elem, vagy egyáltalán nem szerepel a sorozat elemei között.

5. Összefoglalás

„Vajon, ha egy diák nem tud úgy tanulni, ahogy tanítják, nem a tanárnak kellene úgy tanítani, hogy azt a diák is megértse?” Sokszor elgondolkodunk Ignacio Estrada gondolatán, hiszen egy olyan oktatási technika kifejlesztése lenne a cél, amely kimagasló eredményeket érne el a diákok teljesítményét tekintve. Függetlenül a korosztálytól, a nemtől, a szakiránytól szeretnénk mindenki számára egy elérhető platformot biztosítani, ahol elsajátíthatók az algoritmikával kapcsolatos tudnivalók. Az AlgoRythmics oktatási környezet segít elsősorban a tanároknak, hiszen lehetőséget biztosít lecketervek dinamikus létrehozására, melye különböző tanulási lépések révén vezetik el a felhasználót a megértésig. Emellett, segítséget nyújt a felhasználóknak, hiszen mindenki úgy tanulhat, ahogy az számára a legkedvezőbb és legeredményesebb.

Mindenki más és más, és mindenki megérdemli, hogy tanuljon és tanítsák. Ez az E-learning oktatási környezet, annak köszönhetően, hogy az interaktivitás különböző szintjein nyilvánul meg, és a tanulási lépések különböző változatait használja az oktatásban, lehetőséget biztosít arra, hogy mindenki megtalálja a számára leghatékonyabb tanulási módszert.

Célunk, hogy minél inkább be tudjuk vonni az oktatási folyamatba a környezet használatát, és hogy ez által elősegítsük az algoritmusok megértésében való előrehaladást. 2019. májusától kezdve egyetemünkön már elérhetővé vált, és azóta közel 374 tanuló használta. Jelenleg hetente alkalmazzuk tanórákon, melyek során blended – learning oktatási formával tanítjuk diákjainkat és segítjük az algoritmusok megértését (**15. ábra: Regisztrált felhasználók száma szakok szerint**).



15. ábra: Regisztrált felhasználók száma szakok szerint

Irodalom

1. Katai, Z. (2014, June). Selective hiding for improved algorithmic visualization. In Proceedings of the 2014 conference on Innovation & technology in computer science education (pp. 33-38). ACM.
2. Byrne, M. D., Catrambone, R., & Stasko, J. T. (1996). Do algorithm animations aid learning?. Georgia Institute of Technology.

3. Korhonen, A., & Malmi, L. (2000). Algorithm simulation with automatic assessment. ACM SIGCSE Bulletin, 32(3), 160-163.
4. Naps, T. L., Eagan, J. R., & Norton, L. L. (2000, May). JHAVÉ—an environment to actively engage students in Web-based algorithm visualizations. In ACM SIGCSE Bulletin (Vol. 32, No. 1, pp. 109-113). ACM.
5. Jarc, D. J., Feldman, M. B., & Heller, R. S. (2000). Assessing the benefits of interactive prediction using web-based algorithm animation courseware. ACM SIGCSE Bulletin, 32(1), 377-381.