A Prog2 tematika tartalmi megtöltése

A jelen dokumentumban konkrét labor és otthoni feladatokat rendelünk a (számunkra hivatalból előírt, betartandó) heti bontású tematika tételeihez. A heti labormunka az adott héthez rendelt feladatok teljesítésén alapszik. A hallgatók a laboron önállóan dolgoznak, az oktató rövid indító iránymutatása (például a feladatok pontosítása) után, ám a feladatokon otthon is dolgozni kell.

- Egy „feladatcsokor” (egy csokorból 5 feladatot kell választani, lásd például később a „Helló, Arroway!” egy csokor) feladatok teljesítésére időben az adott hét áll rendelkezésre.
    - Az induláskor alkalmazunk egy 2 hetes időbeli puffert, a szeptember 16-i laborodon a „Helló, Berners-Lee!” csokor megkezdését kell bemutatnod, a következő héttől pedig folytatólagosan a „Helló, Arroway!”-től kezdve. Ha adott laborközösségnek laborja elmarad, akkor ez a „hol tartunk számlálót” sem léptetjük.
    - Az a feladat fogadható el megkezdettnek, amelyről részletesen tudósít a hallgató a jegyzőkönyv pdf-jében (vagy magáról a feladatról, ha kész, vagy a nehézségről, amibe beleütközött és gátolja a megoldásban).
    - A jegyzőkönyvet a Prog1-es DocBook jegyzőkönyv második részeként (tehát ugyanabban, annak továbbfejlesztéseként) kell megvalósítani (tehát egy másik part elemben).

Közös munka

Minden labormérést lehet egy a félév során fix, két fős csoportban is végezni. Ennek előkészülete, hogy a két hallgató a prog1-es jegyzőkönyvéből minden prog1-es feladatból kiválasztja a jobb megoldást, ezt teszi be az új mérési jegyzőkönyvbe és a jelen prog2 részt ebben az anyagban valósítja meg. Az új feladatoknál szerepeltetni kell egy bekezdést, ami rögzíti, ki mit csinált a feladatban. Védeni egyénileg kell.

Értékelési szempontok

A jegyzőkönyvnek nyilvános repóban (és DocBook XML 5 forrásokban is) elérhetőnek kell lennie. Ha a DocBook források nem validak, vagy plágium van bennük, akkor az értékelés automatikusan elégtelen.

A gyakorlati jegyet az utolsó laborokon a laborjegyzőkönyvre és a védésre adjuk.

- A jeles szükséges (de nem elégséges) feltétele, hogy
    - minden héten mind az 5 feladatra legyen megoldásunk bemutatva a jegyzőkönyvben. Legalább egy feladat megoldása, annak elmagyarázása, bemutatása legyen kistreamve vagy legalább YB videóban kitéve.
    - Legalább 5 hallgatótárs feltünteti a jegyzőkönyvében, hogy a szóban forgó hallgató tutorja volt vagy volt legalább 50 megnézés videón.
- A jó szükséges (de nem elégséges) feltétele, hogy minden héten az 5 feladatból legyen 4 feladatra megoldásunk bemutatva a jegyzőkönyvben.
- A közepes szükséges (de nem elégséges) feltétele, hogy minden héten az 5 feladatból legyen 3 feladatra megoldásunk bemutatva a jegyzőkönyvben.
- A 9 heti bontásból 1-nél lehet három feladatnál kevesebb megoldás, ha 2 vagy több olyan hét van, ahol három feladatnál kevesebb megoldás van, az automatikusan elégtelen gyakorlati jegyet eredményez.
- Ha bármely hétnél 1 megoldás van vagy egyetlen megoldás sincs, az elégtelen gyakorlati jegyet eredményez.

A félév utolsó laborjait a védésnek szenteljük, amely a jegyzőkönyvből az oktató által kiválasztott feladat gép melletti bemutatásából áll. A jegyzőkönyv és a védés alapján adja a laborvezető a gyakorlati jegyet. Ezt időben támogatandó a tematika néhány párját összevontuk az alábbiak szerint.

Háttér

Minden feladatot megcsináltam már, ezeket eléred az UDPROG közösségben (vagy a repóban, vagy az évkönyvben vagy a fészes csoportban) vagy egyéb célrepókban.

- A zöld, piros és kék (lásd később) feladatok nem kötelezőek.
    - A „deprecated” (zöld) feladatokat is lehet választani, de azok mivel régiek, kevés a támogatottság, tipikusan nehézséget okozhat a megoldásuk… (mert például a felhasznált API-k sokat változtak és már nem gondoztam a kódokat…). De olyan is van, melyet már meghaladtál, például az első részben is feldolgozhattál…
    - A piros feladat kidolgozásával az egész csokor kiváltható, ezek tipikusan kutatási feladatok, beszállhatsz velük akár kutatási kéziratokba társszerzőként, akár TDK-zhatsz, szakdolgozhatsz belőlük.
    - A kék olyan mint a piros, de a tartalmazó és egy másik csokor is kiváltható vele.

A feladatcsokrok

0. hét - „Helló, Berners-Lee!”

A szokásos olvasónapló feladat:

- C++: Benedek Zoltán, Levendovszky Tihamér Szoftverfejlesztés C++ nyelven
- Java: Nyékyné Dr. Gaizler Judit et al. Java 2 útikalauz programozóknak 5.0 I-II.
    - Ebből a két könyvből pár oldalas esszé jellegű kidolgozást kérek, Java és C++ összehasonlítás mentén, pl. kb.: kifejezés fogalom ua., Javában minden objektum referencia, mindig dinamikus a kötés, minden függvény virtuális, klónozás stb.
- Python: Forstner Bertalan, Ekler Péter, Kelényi Imre: Bevezetés a mobilprogramozásba. Gyors prototípus-fejlesztés Python és Java nyelven (35-51 oldal)
    - Itt a kijelölt oldalakból egy 1 oldalas élmény-olvasónaplóra gondoltam.

1. hét - „Helló, Arroway!”

1. hét Az objektumorientált paradigma alapfoglalmai. Osztály, objektum, példányosítás.

OO szemlélet

A módosított polártranszformációs normális generátor beprogramozása Java nyelven. Mutassunk rá, hogy a mi természetes saját megoldásunk (az algoritmus egyszerre két normálist állít elő, kell egy példánytag, amely a nem visszaadottat tárolja és egy logikai tag, hogy van-e tárolt vagy futtatni kell az algot.) és az OpenJDK, Oracle JDK-ban a Sun által adott OO szervezés ua.!

https://arato.inf.unideb.hu/batfai.norbert/UDPROG/deprecated/Prog1_5.pdf (16-22 fólia)

Ugyanezt írjuk meg C++ nyelven is! (lásd még UDPROG repó: source/labor/polargen)

Homokózó

Írjuk át az első védési programot (LZW binfa) C++ nyelvről Java nyelvre, ugyanúgy működjön! Mutassunk rá, hogy gyakorlatilag a pointereket és referenciákat kell kiirtani és minden máris működik (erre utal a feladat neve, hogy Java-ban minden referencia, nincs választás, hogy mondjuk egy attribútum pointer, referencia vagy tagként tartalmazott legyen).

Miután már áttettük Java nyelvre, tegyük be egy Java Servletbe és a böngészőből GET-es kéréssel (például a böngésző címsorából) kapja meg azt a mintát, amelynek kiszámolja az LZW binfáját![1]

„Gagyi”

Az ismert formális[2] „while (x <= t && x >= t && t != x);” tesztkérdéstípusra adj a szokásosnál (miszerint x, t az egyik esetben az objektum által hordozott érték, a másikban meg az objektum referenciája) „mélyebb” választ, írj Java példaprogramot mely egyszer végtelen ciklus, más x, t értékekkel meg nem! A példát építsd a JDK Integer.java forrására[3], hogy a 128-nál inkluzív objektum példányokat poolozza!

[1] Tavalyi prog2 első védés volt.

[2] https://www.facebook.com/groups/udprog/permalink/437825193072042/?comment_id=437862206401674&reply_comment_id=437863669734861&comment_tracking=%7B%22tn%22%3A%22R3%22%7D

[3] A JDK telepítési könyvtárában az src.zip-ben találod.

Yoda

Írjunk olyan Java programot, ami java.lang.NullPointerEx-el leáll, ha nem követjük a Yoda conditions-t! https://en.wikipedia.org/wiki/Yoda_conditions

Kódolás from scratch

Induljunk ki ebből a tudományos közleményből: http://crd-legacy.lbl.gov/~dhbailey/dhbpapers/bbp-alg.pdf és csak ezt tanulmányozva írjuk meg Java nyelven a BBP algoritmus megvalósítását!

Ha megakadsz, de csak végső esetben: https://www.tankonyvtar.hu/hu/tartalom/tkt/javat-tanitok-javat/apbs02.html#pi_jegyei (mert ha csak lemásolod, akkor pont az a fejlesztői élmény marad ki, melyet szeretném, ha átélnél).

2. hét - „Helló, Liskov!”

2. hét Öröklődés, osztályhierarchia. Polimorfizmus, metódustúlterhelés. Hatáskörkezelés. A bezárási eszközrendszer, láthatósági szintek. Absztrakt osztályok és interfészek.

Liskov helyettesítés sértése

Írjunk olyan OO, leforduló Java és C++ kódcsipetet, amely megsérti a Liskov elvet! Mutassunk rá a megoldásra: jobb OO tervezés.

https://arato.inf.unideb.hu/batfai.norbert/UDPROG/deprecated/Prog2_1.pdf (93-99 fólia)
(számos példa szerepel az elv megsértésére az UDPROG repóban, lásd pl. source/binom/Batfai-Barki/madarak/)

Szülő-gyerek

Írjunk Szülő-gyerek Java és C++ osztálydefiníciót, amelyben demonstrálni tudjuk, hogy az ősön keresztül csak az ős üzenetei küldhetőek!

https://arato.inf.unideb.hu/batfai.norbert/UDPROG/deprecated/Prog2_1.pdf (98. fólia)

Anti OO

A BBP algoritmussal[4] a Pi hexadecimális kifejtésének a 0. pozíciótól számított 10^6, 10^7, 10^8 darab jegyét határozzuk meg C, C++, Java és C# nyelveken és vessük össze a futási időket!
https://www.tankonyvtar.hu/hu/tartalom/tkt/javat-tanitok-javat/apas03.html#id561066

deprecated - Hello, Android!

Élesszük fel a https://github.com/nbatfai/SamuEntropy/tree/master/cs projektjeit és vessünk össze néhány egymásra következőt, hogy hogyan változtak a források!

Hello, Android!

Élesszük fel az SMNIST for Humans projektet!
https://gitlab.com/nbatfai/smnist/tree/master/forHumans/SMNISTforHumansExp3/app/src/main
Apró módosításokat eszközölj benne, pl. színvilág.

Hello, SMNIST for Humans!

Fejleszd tovább az SMNIST for Humans projektet SMNIST for Anyone emberre szánt appá! Lásd az smnist2_kutatasi_jegyzokonyv.pdf-ben a részletesebb hátteret!

Ciklomatikus komplexitás

Számoljuk ki valamelyik programunk függvényeinek ciklomatikus komplexitását! Lásd a fogalom tekintetében a https://arato.inf.unideb.hu/batfai.norbert/UDPROG/deprecated/Prog2_2.pdf (77-79 fóliát)!

[4] https://www.tankonyvtar.hu/hu/tartalom/tkt/javat-tanitok-javat/apbs02.html#pi_jegyei

3. hét - „Helló, Mandelbrot!”

3. hét Modellező eszközök és nyelvek. AZ UML és az UML osztálydiagramja.

Reverse engineering UML osztálydiagram

UML osztálydiagram rajzolása az első védési C++ programhoz. Az osztálydiagramot a forrásokból generáljuk (pl. Argo UML, Umbrello, Eclipse UML) Mutassunk rá a kompozíció és aggregáció kapcsolatára a forráskódban és a diagramon, lásd még: https://youtu.be/Td_nlERlEOs.

https://arato.inf.unideb.hu/batfai.norbert/UDPROG/deprecated/Prog1_6.pdf (28-32 fólia)

Forward engineering UML osztálydiagram

UML-ben tervezzünk osztályokat és generáljunk belőle forrást!

Egy esettan

A BME-s C++ tankönyv 14. fejezetét (427-444 elmélet, 445-469 az esettan) dolgozzuk fel!

BPMN

Rajzoljunk le egy tevékenységet BPMN-ben!
https://arato.inf.unideb.hu/batfai.norbert/UDPROG/deprecated/Prog2_7.pdf (34-47 fólia)

BPEL Helló, Világ! - egy visszhang folyamat

Egy visszhang folyamat megvalósítása az alábbi teljes „videó tutoriál” alapján:
https://youtu.be/0OnlYWX2v_I

TeX UML

Valamilyen TeX-es csomag felhasználásával készíts szép diagramokat az OOCWC projektről (pl. use case és class diagramokat).

4. hét - „Helló, Chomsky!”

4. hét Objektumorientált programozási nyelvek programnyelvi elemei: karakterkészlet, lexikális egységek, kifejezések, utasítások.

Encoding

Fordítsuk le és futtassuk a Javat tanítok könyv MandelbrotHalmazNagyító.java forrását úgy, hogy a fájl nevekben és a forrásokban is meghagyjuk az ékezetes betűket!
https://www.tankonyvtar.hu/hu/tartalom/tkt/javat-tanitok-javat/adatok.html

OOCWC lexer

Izzítsuk be az OOCWC-t és vázoljuk a https://github.com/nbatfai/robocar-emulator/blob/master/justine/rcemu/src/carlexer.ll lexert és kapcsolását a programunk OO struktúrájába!

l334d1c4[5]

Írj olyan OO Java vagy C++ osztályt, amely leet cipherként működik, azaz megvalósítja ezt a betű helyettesítést: https://simple.wikipedia.org/wiki/Leet (Ha ez első részben nem tetted meg, akkor írasd ki és magyarázd meg a használt struktúratömb memóriafoglalását!)

Full screen

Készítsünk egy teljes képernyős Java programot!
Tipp: https://www.tankonyvtar.hu/en/tartalom/tkt/javat-tanitok-javat/ch03.html#labirintus_jatek

Paszigráfia Rapszódia OpenGL full screen vizualizáció

Lásd vis_prel_para.pdf! Apró módosításokat eszközölj benne, pl. színvilág, textúrázás, a szintek jobb elkülönítése, kézreállóbb irányítás.

Paszigráfia Rapszódia LuaLaTeX vizualizáció

Lásd vis_prel_para.pdf! Apró módosításokat eszközölj benne, pl. színvilág, még erősebb 3D-s hatás.

Perceptron osztály

Dolgozzuk be egy külön projektbe a projekt Perceptron osztályát!
Lásd https://youtu.be/XpBnR31BRJY

[5]Lásd a C+lex megoldásom itt:
https://www.facebook.com/groups/udprog/permalink/942314465956443/?comment_id=942571282597428&comment_tracking=%7B%22tn%22%3A%22R3%22%7D

5. hét - „Helló, Stroustrup!”

5. hét Objektumorientált programozási nyelvek típusrendszere (pl.: Java, C#) és 6. hét Típusok tagjai: mezők, (nevesített) konstansok, tulajdonságok, metódusok, események, operátorok, indexelők, konstruktorok, destruktorok, beágyazott típusok.

Összevonva.

JDK osztályok

Írjunk olyan Boost C++ programot (indulj ki például a fénykardból) amely kilistázza a JDK összes osztályát (miután kicsomagoltuk az src.zip állományt, arra ráengedve)!

Másoló-mozgató szemantika

Kódcsipeteken (copy és move ctor és assign) keresztül vesd össze a C++11 másoló és a mozgató szemantikáját, a mozgató konstruktort alapozd a mozgató értékadásra!

Hibásan implementált RSA törése

Készítsünk betű gyakoriság alapú törést egy hibásan implementált RSA kódoló: https://arato.inf.unideb.hu/batfai.norbert/UDPROG/deprecated/Prog2_3.pdf (71-73 fólia) által készített titkos szövegen.

Változó argumentumszámú ctor

Készítsünk olyan példát, amely egy képet tesz az alábbi projekt Perceptron osztályának bemenetére és a Perceptron ne egy értéket, hanem egy ugyanakkora méretű „képet” adjon vissza. (Lásd még a 4 hét/Perceptron osztály feladatot is.)

Összefoglaló

Az előző 4 feladat egyikéről írj egy 1 oldalas bemutató „”esszé szöveget!

6. hét - „Helló, Gödel!”

7. hét Interfészek. Kollekciók. és 8. hét Funkcionális nyelvi elemek. Lambda kifejezések.

Összevonva.

Gengszterek

Gengszterek rendezése lambdával a Robotautó Világbajnokságban
https://youtu.be/DL6iQwPx1Yw (8:05-től)

C++11 Custom Allocator

https://prezi.com/jvvbytkwgsxj/high-level-programming-languages-2-c11-allocators/ a CustomAlloc-os példa, lásd C forrást az UDPROG repóban!

STL map érték szerinti rendezése

Például: https://github.com/nbatfai/future/blob/master/cs/F9F2/fenykard.cpp#L180

Alternatív Tabella rendezése

Mutassuk be a https://progpater.blog.hu/2011/03/11/alternativ_tabella a programban a java.lang Interface Comparable<T> szerepét!

Prolog családfa

Ágyazd be a Prolog családfa programot C++ vagy Java programba! Lásd para_prog_guide.pdf!

GIMP Scheme hack

Ha az előző félévben nem dolgoztad fel a témát (például a mandalás vagy a króm szöveges dobozosat) akkor itt az alkalom!

7. hét - „Helló, !”

9. hét Adatfolyamok kezelése, streamek és 11. hét I/O, állománykezelés. Szerializáció.

Összevonva.

FUTURE tevékenység editor

Javítsunk valamit a ActivityEditor.java JavaFX programon!
https://github.com/nbatfai/future/tree/master/cs/F6
Itt láthatjuk működésben az alapot: https://www.twitch.tv/videos/222879467

OOCWC Boost ASIO hálózatkezelése

Mutassunk rá a scanf szerepére és használatára! https://github.com/nbatfai/robocar-emulator/blob/master/justine/rcemu/src/carlexer.ll

SamuCam

Mutassunk rá a webcam (pl. Androidos mobilod) kezelésére ebben a projektben: https://github.com/nbatfai/SamuCam

BrainB

Mutassuk be a Qt slot-signal mechanizmust ebben a projektben: https://github.com/nbatfai/esport-talent-search

OSM térképre rajzolása[6]

Debrecen térképre dobjunk rá cuccokat, ennek mintájára, ahol én az országba helyeztem el a DEAC hekkereket: https://www.twitch.tv/videos/182262537 (de az OOCWC Java Swinges megjelenítőjéből: https://github.com/nbatfai/robocar-emulator/tree/master/justine/rcwin is kiindulhatsz, mondjuk az komplexebb, mert ott időfejlődés is van...)

[6]Alternatívaként készíthetsz egy GoogleMaps alapú Androidos „GPS trackert”, 2007 óta csinálok ilyen példát: https://youtu.be/QStgBZ6JfAU az aktuális a Bátfai Haxor Stream keretében: https://bhaxor.blog.hu/2018/09/19/nandigps_ismerkedes_a_gps-el

8. hét - „Helló, Lauda!”

10. hét Kivételkezelés. és 12. hét Reflexió. A fordítást és a kódgenerálást támogató nyelvi elemek (annotációk, attribútumok).

Összevonva.

Port scan

Mutassunk rá ebben a port szkennelő forrásban a kivételkezelés szerepére!

https://www.tankonyvtar.hu/hu/tartalom/tkt/javat-tanitok-javat/ch01.html#id527287

AOP

Szőj bele egy átszövő vonatkozást az első védési programod Java átiratába! (Sztenderd védési feladat volt korábban.)

Android Játék

Írjunk egy egyszerű Androidos „játékot”! Építkezzünk például a 2. hét „Helló, Android!” feladatára!

Junit teszt

A https://progpater.blog.hu/2011/03/05/labormeres_otthon_avagy_hogyan_dolgozok_fel_egy_pedat poszt kézzel számított mélységét és szórását dolgozd be egy Junit tesztbe (sztenderd védési feladat volt korábban).

OSCI

Készíts egyszerű C++/OpenGL-es megjelenítőt, amiben egy kocsit irányítasz az úton.

OSCI2

Készíts egyszerű C++/OpenGL-es megjelenítőt, amiben egy kocsit irányítasz az úton. A kocsi állapotát minden pillanatban mentsd le. Ezeket add át egy Prolog programnak, ami egyszerű reflex ágensként adjon vezérlést a kocsinak, hasonlítsd össze a kézi és a Prolog-os vezérlést. Módosítsd úgy a programodat, hogy ne csak kézzel lehessen vezérelni a kocsit, hanem a Prolog reflex ágens vezérelje!

OSCI3

Készíts egy OSM utakat megjelenítő C++/OpenGL-es progit!

9. hét - „Helló, Calvin!”

13. hét Multiparadigmás nyelvek és 14. hét Programozás multiparadigmás nyelveken.

Összevonva.

MNIST

Az alap feladat megoldása, +saját kézzel rajzolt képet is ismerjen fel, https://progpater.blog.hu/2016/11/13/hello_samu_a_tensorflow-bol Háttérként ezt vetítsük le: https://prezi.com/0u8ncvvoabcr/no-programming-programming/

Deep MNIST

Mint az előző, de a mély változattal. Segítő ábra, vesd össze a forráskóddal a https://arato.inf.unideb.hu/batfai.norbert/NEMESPOR/DE/denbatfai2.pdf 8. fóliáját!

CIFAR-10

Az alap feladat megoldása, +saját fotót is ismerjen fel, https://progpater.blog.hu/2016/12/10/hello_samu_a_cifar-10_tf_tutorial_peldabol

Android telefonra a TF objektum detektálója

Telepítsük fel, próbáljuk ki!

SMNIST for Machines

Készíts saját modellt, vagy használj meglévőt, lásd: https://arxiv.org/abs/1906.12213

Minecraft MALMO-s példa

A https://github.com/Microsoft/malmo felhasználásával egy ágens példa, lásd pl.: https://youtu.be/bAPSu3Rndi8, https://bhaxor.blog.hu/2018/11/29/eddig_csaltunk_de_innentol_mi, https://bhaxor.blog.hu/2018/10/28/minecraft_steve_szemuvege

Debrecen, 2019-9-7
Dr. Bátfai Norbert