

Eötvös Loránd Tudomány Egyetem

Informatikai kar

## Numerikus Analízis Tanszék

##### Oktatási keretrendszer interpolációs feladatok megoldásához

##### Témavezető: Szerző:

## Fábián Gábor Bárczay Kristóf

##### Budapest, 2017

Tartalom

[1. Bevezető 3](#_Toc500683943)

[2. Felhasználói dokumentáció 4](#_Toc500683944)

[2.1. Feladat 4](#_Toc500683945)

[2.2. Telepítési útmutató 4](#_Toc500683946)

[2.3. Használati útmutató 5](#_Toc500683947)

[3. Fejlesztői dokumentáció 12](#_Toc500683948)

[3.1. A feladat és a hozzá választott programozási nyelv 12](#_Toc500683949)

[3.2. A fejlesztés során használt eszközök 12](#_Toc500683950)

## Bevezető

Szakdolgozatom témája egy házi feladat ellenőrző, tanulást segítő program implementálása, amely elsősorban az ELTE Informatika karán meghirdetett Numerikus módszerek 2. tárgyhoz nyújt támogatást a hallgatóknak, de természetesen mindenkinek hasznos lehet, aki érdeklődik a matematika ezen területe iránt.

Az egyetemen tanár szakon is tanulmányokat folytatok, az ott szerzett pedagógiai és a programtervező informatikus szakon szerzett matematikai ismereteim, tapasztalataim integrálása motivált egy olyan oktató program elkészítésére, amellyel a tárgyat végző hallgatók munkáját tudom segíteni. Ezen túl, úgy ítéltem meg, hogy nekem magamnak is jó tapasztalatot nyújt a program elkészítése, ha esetleg később, a saját óráimra szeretnék majd készíteni ugyan valamivel könnyedebb, középiskolás szintű feladatokat megoldó, matematikai alkalmazást.

Oktató programról lévén szó a hangsúlyt a segítségre, támogatásra helyeztem, nem arra, hogy képes legyen megoldani a hallgatók helyett a házi feladatokat.

A program segítségével, a fentiekben említett tárgy (Numerikus módszerek 2. ) első felében tanult módszerekkel megoldható feladatokat tudja ellenőrizni a felhasználó. További célként fogalmaztam meg, hogy a feladatok ellenőrzéséhez ne kelljen egy új programozási nyelvet megtanulni, hiszen ezek a feladatok matlab-ban is ellenőrizhetőek, de az ottani ellenőrzéshez a nyelv viszonylag magas ismerete szükséges. A program csupán matematikai ismereteket igényel, és a kevésbé járatos felhasználók számára is támogatást nyújt.

A program, jó segítség lehet azoknak is, akik komolyabban kívánnak foglalkozni az interpolációs polinomok témakörével, ami önmagában is rendkívül érdekes és még igen nagy gyakorlati haszna is van, például a digitális hang, kép illetve audiovizuális tartalmak szerkesztésében hogy néhány példát említsek a sok közül.

## Felhasználói dokumentáció

A dolgozat 2. részében, felhasználói dokumentáció keretében igyekszem részletesen bemutatni a felhasználók számára a program által nyújtott lehetőségeket, valamint a program szakszerű használatát.

### Feladat

A megoldani kívánt probléma, egy olyan alkalmazás elkészítése, amelyben könnyedén eligazodva tudják a felhasználók ellenőrizni a kívánt interpolációs feladatok megoldását.

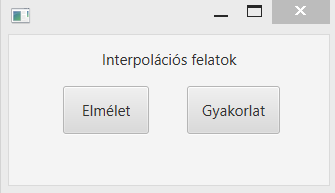
A program alapvetően két nagy egységre bontható, egy elméleti valamint egy gyakorlati egységre.

Az elméleti részben a témához kapcsolódó, rövid áttekintés olvasható vázlatos formában, a saját jegyzeteim alapján. A gyakorlati blokk már komplexebb felület, melynek használatára részletesebben a későbbiekben külön kitérek, de alapvetően itt alappont-érték párokra és alappontok illetve a közelíteni kívánt függvény értékekre lehet, Lagrange, Newton, Inverz, Hermite és Spline interpolációkkal polinomokat illeszteni, valamint a kiszámolt polinomokat koordináta rendszerben is megtekinteni. A két rész jól elkülöníthetően külön ablakban jelenik meg így akár párhuzamosan is lehet tanulni valamint ellenőrizni megoldásokat.

### Telepítési útmutató

/\*…………….\*/

### Használati útmutató

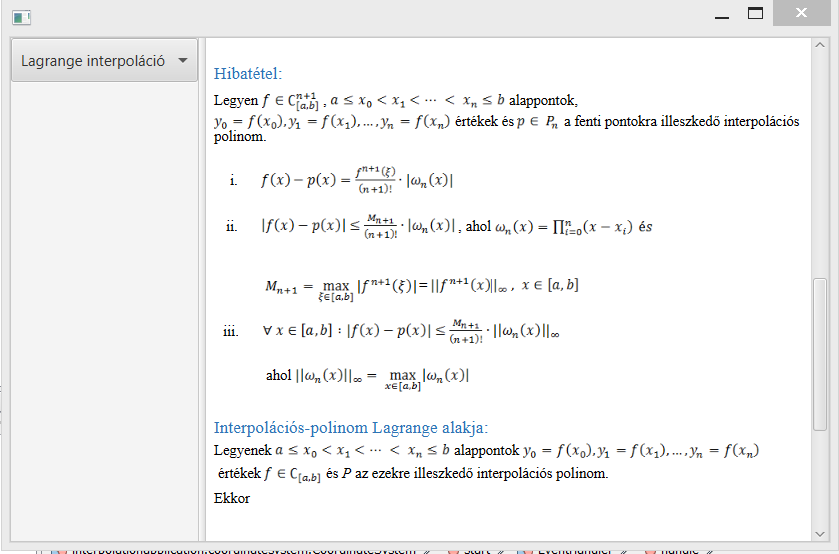
A program indításakor egy ablak jelenik meg *(2.3.-1. ábra),* melyen egyből kiválasztható, hogy elméleti vagy gyakorlati tudásunkat szeretnénk-e bővíteni. A gombokat értelemszerűen lehet használni megfelelő gombra kattintva a megfelelő ablak jelenik meg.

2.3‑1. ábra

#### Elmélet ablak

Nézzük először, mi történik, ha az elmélet gombot választjuk. A gombra való kattintás után, egy új ablak jelenik meg *(2.3-2. ábra)*, mely két részből áll. Az ablak baloldalán látható egy legördülő menü gomb, amelynek segítségével kiválasztható az éppen tanulni kívánt részt.

A jobb oldalon pedig megjelenik a kiválasztott anyagrész, rövid összefoglalója. Az elméleti leírások, olykor nehezebb matematikai levezetéseket, tételeket és bizonyításokat tartalmaznak, ezért ahol lehet,az oldal végén rövid összefoglalás segíti a felhasználót abban, hogy miről is szól az adott elméleti anyagrész. Természetesen a közérthetőség volt a cél, de olyan mélységekbe nem merültem, hogy minden egyes matematikai jelölés illetve művelet magyarázata külön megjelenítésre kerüljön. Jelenleg a program azokat az elméleti anyagrészeket tartalmazza, amelyek a gyakorlati részben is szerepelnek, de idővel igyekszem bővíteni és akár az egész tárgy elméleti ismeretanyaga belekerülhet.

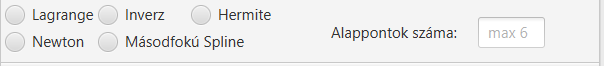


*Menü Elméleti anyag*

2.3‑1. ábra

#### Gyakorlati ablak

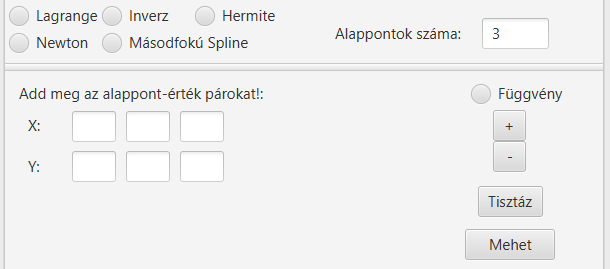
A gyakorlat gomb megnyit egy új ablakot*.*A megjelenés kicsit összetettebb, mint az elmélet résznél, de megítélésem szerint a használata nem túl bonyolult. A fejlécben szerepel öt rádiógomb illetve egy szöveges mező ahova az alappontok számát kell beírni *(2.3-3.ábra).*



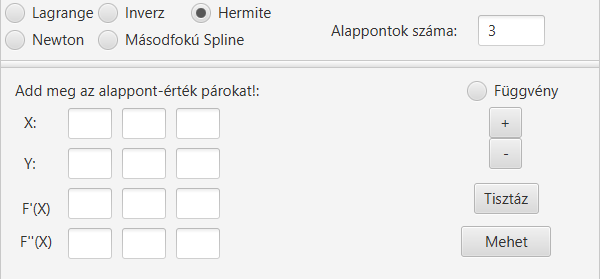
*A választható interpolációk*  *Alappontok száma*

2.3‑2. ábra

Ahogy a fenti ábrán látható *(2.3-3. ábra),* az alappontok száma legfeljebb hat és legalább kettő kell, hogy legyen. A program az elején nem jelzi a minimum kettőt, feltételezve azt, hogy a felhasználó tudja, hogy egy alappontra nincs értelme interpolációs polinomot illeszteni. Ellenben, ha mégis túl kicsi vagy túl nagy számot ír be, vagy esetleg nem is egész szám, amit megadott, akkor egy felugró ablak figyelmeztetni fogja, a helytelen bevitelre. Fontos, hogy miután bekerült a szám a mezőbe, ezután kötelezően entert nyomva megjelennek további mezők, a kiválasztott interpoláció függvényében.



2.3‑4. ábra



2.3‑5. ábra

A 2.3-4. ábra illetve a 2.3.-5. ábra csak két példát mutat a sok közül.

Elsőként az látható, hogy mi történik, ha nem választjuk ki az interpoláció fajtáját. Ebben az esetben a program automatikusan csak az alappont érték párok beírására lehetőséget adó szövegdobozokat jeleníti meg, és a későbbiekben figyelmeztet az interpoláció fajtájának kiválasztására.

A második ábrán a Hermite interpoláció van kiválasztva, ehhez az interpolációhoz pontonként négy érték adható meg. A képernyő bal oldalán látható + illetve gombok segítségével további alappontok vehetőek fel, illetve törölhetőek továbbra is csak kettő és hat között maradva.

Nézzük most az egyes rádiógombok funkcióit.

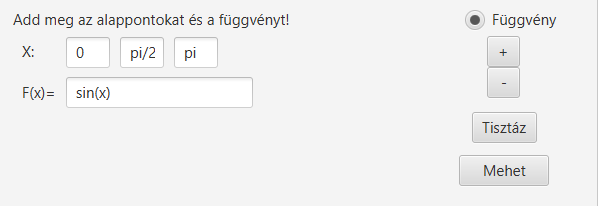
* **Lagrange:** Értelem szerűen az interpolációs polinom Lagrange- alakját számolja ki. Ekkor csak az alappont-érték párokat lehet megadni és minden mező kitöltése kötelező.
* **Newton:** Az interpolációs polinom Newton-alakját számolja ki. A pontok kitöltésében hasonlóan kell eljárni, mint az előző esetben.
* **Inverz:** Az alappont-érték párokon végrehajtja az inverz interpoláció egy lépését, valamint kiszámolja a helyettesítési értéket az helyen. Figyelnünk kell arra, hogy a megadott függvény szigorúan monoton legyen különben nem invertálható.
* **Hermite:** Ezt a gombot kiválasztva négy sor jelenik meg. Az alappont-érték párokon kívül megadható az egyes pontokhoz tartozó első illetve második derivált. Ezen mezők kitöltése nem kötelező, de ha sehol nem adjuk meg a program figyelmeztet, hogy így a Newton interpolációt is használhatnánk.
* **Másodfokú Spline:** Három sor jelenik meg a képernyőn, ahol a harmadik sorban a másodfokú lokális bázisú spline peremfeltételét lehet megadni valamelyik ponthoz. Csak egyetlen peremfeltétel megadása lehetséges.

A műveletek mögötti matematikai eljárások megvalósítása a fejlesztői dokumentációban olvasható.

A fejlécen kívül még egy rádiógomb található „Függvény” felirattal. Ezt az opciót választva alappont-érték párok helyett alappontokat és a közelíteni kívánt függvényt adhatjuk meg. A program a háttérben automatikusan kiszámolja a pontokhoz tartozó függvényértékeket behelyettesítéssel *(2.3 -6. ábra)*. Mint látható bármely mezőbe beírhatunk összetettebb matematikai kifejezéseket a számítógépes matematika szintaktikájának megfelelően, ha valamit nem megfelelően adunk meg a program jelezni fogja.

*Függvény opció kiválasztása*

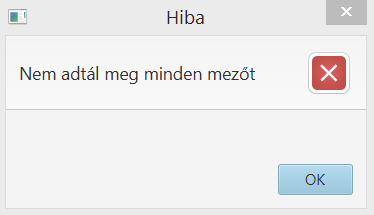
*Alappontok felvétele-törlése*



*Mezők kiürítése Számolás indítása*

2.3‑6. ábra

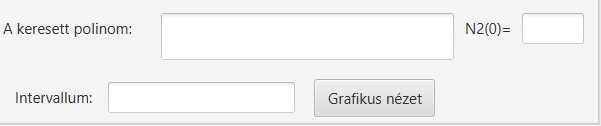
Az eddig említett gombok és mezők mellett látható egy „Tisztáz” és egy „Mehet” feliratú gomb is. Előbbi azt a célt szolgálja, hogy szövegdobozok tartalmát eltávolítja. Ez úgy is elérhető, ha az elején látott „Alappontok száma” mezőbe új értéket adunk meg. A + illetve a gombok használatával minden érték marad a helyén csak az újabban felvett mezőket szükséges kitölteni. A „Mehet” gombbal tudjuk elindítani a számolást. Amikor rákattintunk, megtörténnek az ellenőrzések. Mindent helyesen töltöttünk-e ki és, hogy kiválasztottuk-e az interpoláció fajtáját. Ellenkező esetben egy a 2.3-7. ábrán látható ablak fog megjelenni a megfelelő üzenettel. Ha nem növekvő sorrendben adtuk meg a pontokat, akkor a program automatikusan növekvő sorrendbe teszi.



2.3‑7. ábra

Nézzük a gyakorlati felület harmadik részét, ahol a kiszámolt polinom, spline megtalálható, illetve koordináta rendszerben megtekinthető.

*A kiszámított polinom helye Inverz esetén a 0-ban vett érték*

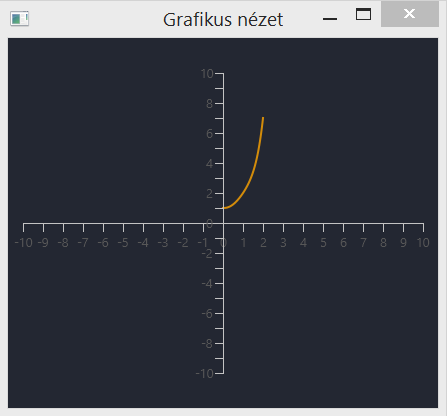
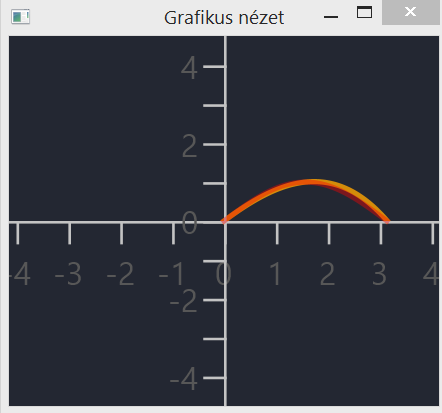


*Az ábrázolni kívánt intervallum Koordináta rendszer megnyitása*

2.3‑8. ábra

A 2.3-8. ábrán látható a mezők illetve a gombok funkciói. A felhasználói dokumentáció végén meg lehet tekinteni egy teljesen kitöltött felületet is, egyelőre itt nézzük mi mire való. Értelemszerűen „A keresett polinom:” részbe kerül az általunk megadott feltételeknek megfelelő interpolációs polinom illetve spline. A pontok megadása után az „Intervallum” mező automatikusan kitöltődik a „Mehet” gombra való kattintáskor. Az intervallum bal végpontja a legkisebb, míg jobb végpontja a legnagyobb alappont lesz. Ha tágabb intervallumon szeretnénk megtekinteni a kiszámolt függvényt, természetesen megtehetjük, az intervallum helyes megadásával. Inverz interpoláció kiválasztása esetén a polinom nullában vett helyettesítési értékét is kiszámolja. Ez az érték jelenik meg az „N2(0)=” mezőben. A „Grafikus nézet” gombra kattintva megjelenik egy újabb ablak egy koordináta rendszerrel és az ábrázolt polinommal illetve spline-nal *(2.3.-9 és 2.3-10. ábra).* A koordináta rendszer tízszer tízes, mivel ennél nagyobb, illetve kisebb pontok nem igazán fordulnak elő a tanulás során. A 2.3.-10. ábrán két függvény látható. Ilyen eset akkor fordulhat elő, ha korábban a függvény rádió gombot bejelöltük. Ebben az esetben megjelenik az eredetileg megadott függvény piros színnel, és megjelenik a kiszámolt interpolációs polinom narancssárga színnel. A program a hibabecsléseket nem számolja, lévén ahhoz olyan sokat kell deriválni, hogy annak programozása egy újabb diplomamunka feladatát jelentené. Ebből kiindulva úgy gondoltam, hogy látványosabb, ha megjelenítem az eredeti függvényt is, és így szabad szemmel is látható lesz az eltérés, nem kell hozzá képletek sokaságát kiszámolni. Természetesen a hiba pontos értékének kiszámítása nagyon fontos, de ezen számítások elvégzését a felhasználóra bízzuk.

A koordináta rendszerben lehet közelíteni az egér görgőjének segítségével, valamint rákattintva az ablakra arrébb is tudjuk húzni azt, így igazán közelről meg tudjuk tekinteni a kiszámolt függvényeket.



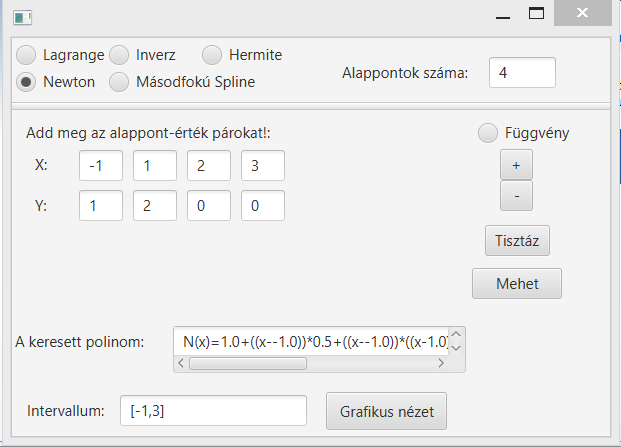
2.3‑9. ábra

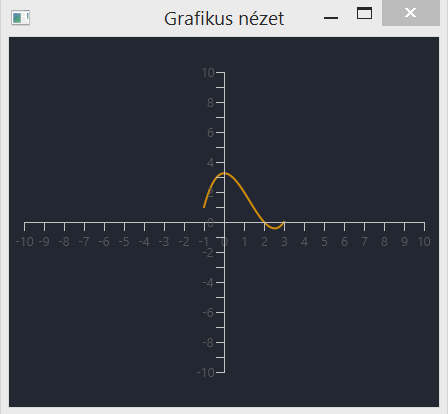
2.3‑10. ábra   
(sin(x) )

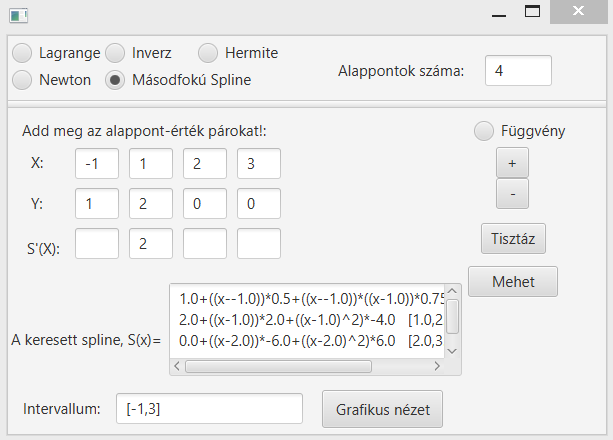
A fentiekben már jeleztem, hogy a program figyeli a hibásan bevitt adatokat. Most szeretném egyben összegyűjteni ezek közül a legfontosabbakat, megkímélve a felhasználót a hibaüzenetektől:

* Figyeljünk oda, hogy szintaktikailag helyesen adjuk meg a matematikai kifejezéseket, az intervallumot, illetve a számokat!
* Az alappont-érték párokat minden mezőben adjuk meg!
* Mindig válasszuk ki az interpoláció fajtáját!
* Figyeljünk arra, hogy a kiválasztott interpolációnak megfelelően töltsük ki a mezőket, ha valamiben nem vagyunk biztosak, tekintsük át az elméleti részben!
* A program a derivált értékeket függvény esetén nem ellenőrzi, ezért ezeket gondosan számoljuk ki!
* Figyeljünk az alappontok sorrendjére illetve inverz interpoláció esetén a monotonitásra!

Végül nézzük meg a program néhány lehetséges kimenetelét:







## Fejlesztői dokumentáció

A fejlesztői dokumentációban, a feladat megvalósításának menetét szeretném bemutatni. A programozási nyelvet, a használt eszközöket, a program felépítést valamint az általam megírt algoritmusokat.

### A feladat és a hozzá választott programozási nyelv

Most a feladatot nem felhasználói, hanem programozói oldalról szeretném egy kicsit bemutatni. A feladat megoldásához mindenképpen egy olyan programozási nyelvet kellett választanom, ami lehetőséget biztosít grafikus felületű alkalmazások fejlesztésére. A program hátterében komplex matematikai algoritmusok vannak ezért szükséges volt, hogy a nyelv tudjon kezelni függvényeket valamint összetett matematikai kifejezéseket. A feladat egy többrétegű alkalmazás implementálását igényelte, külön választva a grafikus felületet, a felület irányítását, valamint az egészhez tartozó logikát. Igyekeztem ezeket a részeket jól elkülönítve, átlátható módon kezelni, ügyelve arra, hogy a program a későbbiekben könnyen karbantartható illetve továbbfejleszthető legyen. Az itt felsoroltakat figyelembe véve esett a választásom a JavaFX nyelvre.

#### A programozási nyelv

Mivel a JavaFX még nem igazán nevezhető elterjedtnek ezért úgy gondoltam, hogy egy pár gondolatban bemutatnám a sajátosságait. A JavaFX tulajdonképpen grafikus és média csomagok halmaza, mellyel könnyedén lehet kliens alkalmazásokat fejleszteni. A dokumentáció az alábbi linken elérhető (2017.12.10-es link):

<https://docs.oracle.com/javafx/2/overview/jfxpub-overview.htm>

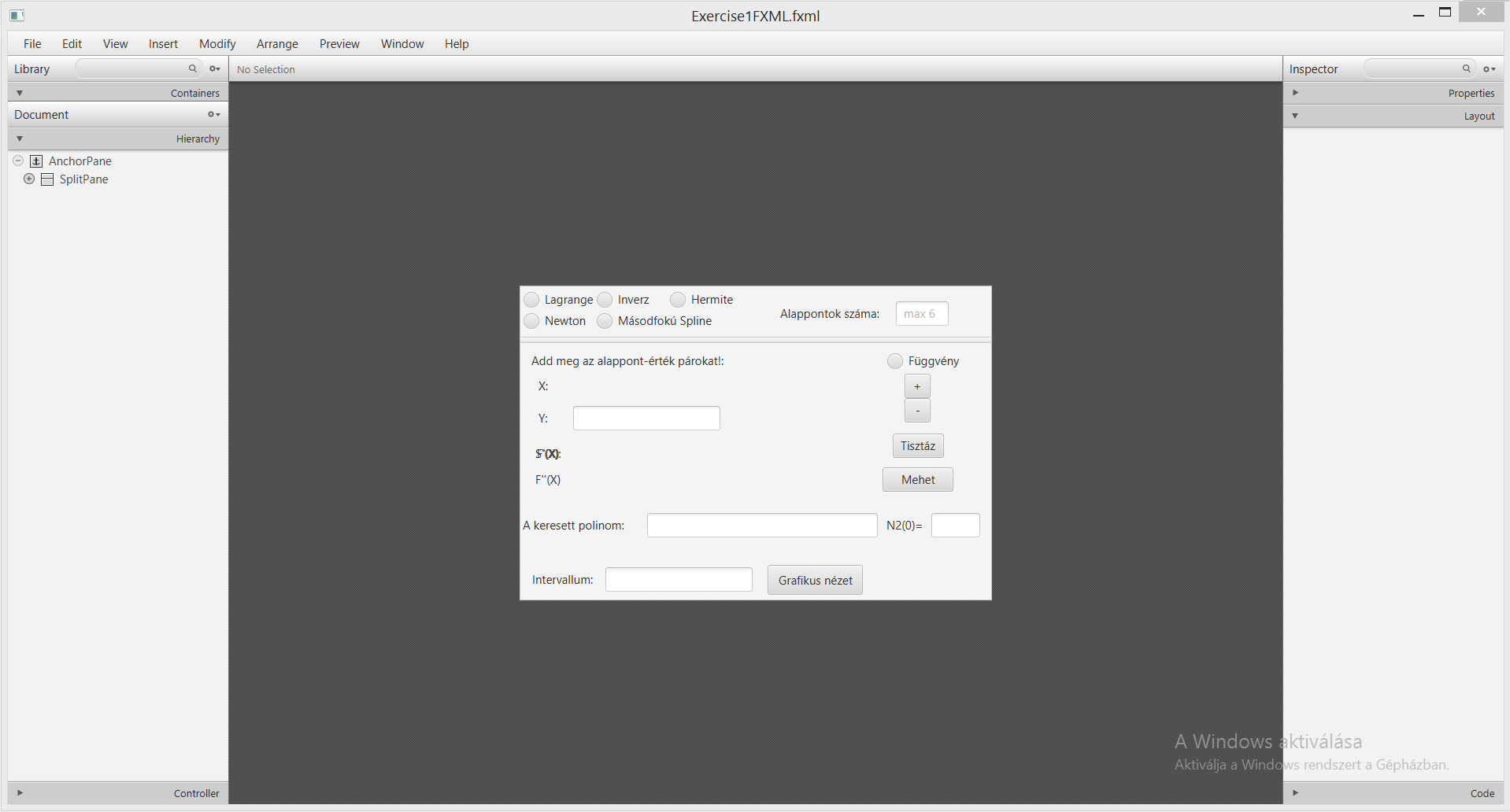
A megközelítés tulajdonképpen nevezhető az általam az egyetemen is tanult Java Swing utódjának, csak korszerűbb és a lehetőségek tárháza is nagyobb. Az egésznek az alapja az egyes ablakokhoz tárolt fxml kiterjesztésű állományok, melyekben megtalálható a felhasználói felület formális leírása. A leírás tulajdonképpen egy xml állomány fa-szerű szerkezettel. Itt található meg minden gomb, szövegdoboz valamint felület leírása, mérete, színe elhelyezkedése stb. A program az adatokat innen kiolvasva állítja fel a felhasználói felületet. A programkönyvtár, akárcsak a Java, az Oracle céghez köthető és Java nyelvi alapokon fekszik. A legfrissebb kiadás a JavaFX 2, jómagam is ezt használtam, de a számból látszik, hogy a technológia még igencsak gyerekcipőben jár és vannak is hiányosságai, például nehezen unit tesztelhető, de a feladatom megvalósításához tökéletesen megfelelőnek találtam.

### A fejlesztés során használt eszközök

A program fejlesztése során a NetBeans fejlesztő környezetet használtam. Úgy gondolom, hogy a fejlesztő környezetek nagyon jól könnyítik és segítik az ember munkáját, ezért elszakadtam a parancssori fordító használatától. Több fejlesztő környezetet kipróbáltam már (Eclipse, IntelliJ Idea), de NetBeans-ben tudok leginkább otthonosan mozogni ezért esett erre a választásom.

Továbbá a program mögött egy build rendszer is található amit az Apache Maven valósít meg. A szoftver a build folyamat automatizálására használható. Lefordítja az állományokat, futtatja a teszteket és a végén az egész alkalmazást egy jar fájlba tömöríti, aminek a segítségével azonnal indítható a program csupán egy Java Runtime Environment szükséges hozzá. Minden projekthez tartozik egy a projektet leíró fájl mely egységesen a pom.xml. Nagyon hasznos még, hogy a mások által készített programcsomagokat nagyon egyszerűen lehet a saját projektünkhöz adni. Elegendő a Maven projekthez tartozó pom.xml állományban függőségként beállítani és már használhatjuk is kívánt osztályokat.

A fejlesztés során verzió kezelő rendszert is használtam még pedig a GitHub-ot. Fontosnak tartottam, hogy ha bármi történik, a programom mindig a legfrissebb állapotában legyen könnyen elérhető és erre a legalkalmasabbnak ezt a rendszert találtam.

A grafikus felület szerkesztéséhez használtam továbbá egy JavaFX SceneBuilder 2.0 nevű szoftvert. Ez segítséget nyújt a grafikus felület megtervezésében és az itt megtervezett ablakokhoz automatikusan legenerálja a hozzájuk tartozó fxml kiterjesztésű állományt.

3.2‑1. ábra  
JavaFX SceneBuilder