Varga Mátyás

|  |  |
| --- | --- |
| INBPM0315-17 | Magas szintű programozási nyelvek 2 |

1.hét

**OO szemlélet** :

.Egységbezárás,adatrejtés,öröklés:

Java: <https://github.com/megvagyhadnagy01/MagasProg2/tree/master/1_gyak/OO%20szeml%C3%A9let/polarJava>

C++:

<https://github.com/megvagyhadnagy01/MagasProg2/tree/master/1_gyak/OO%20szeml%C3%A9let/polargenC%2B%2B>

------------------------------------------------------------------------------------------------------------------------->

**Homokozó:**

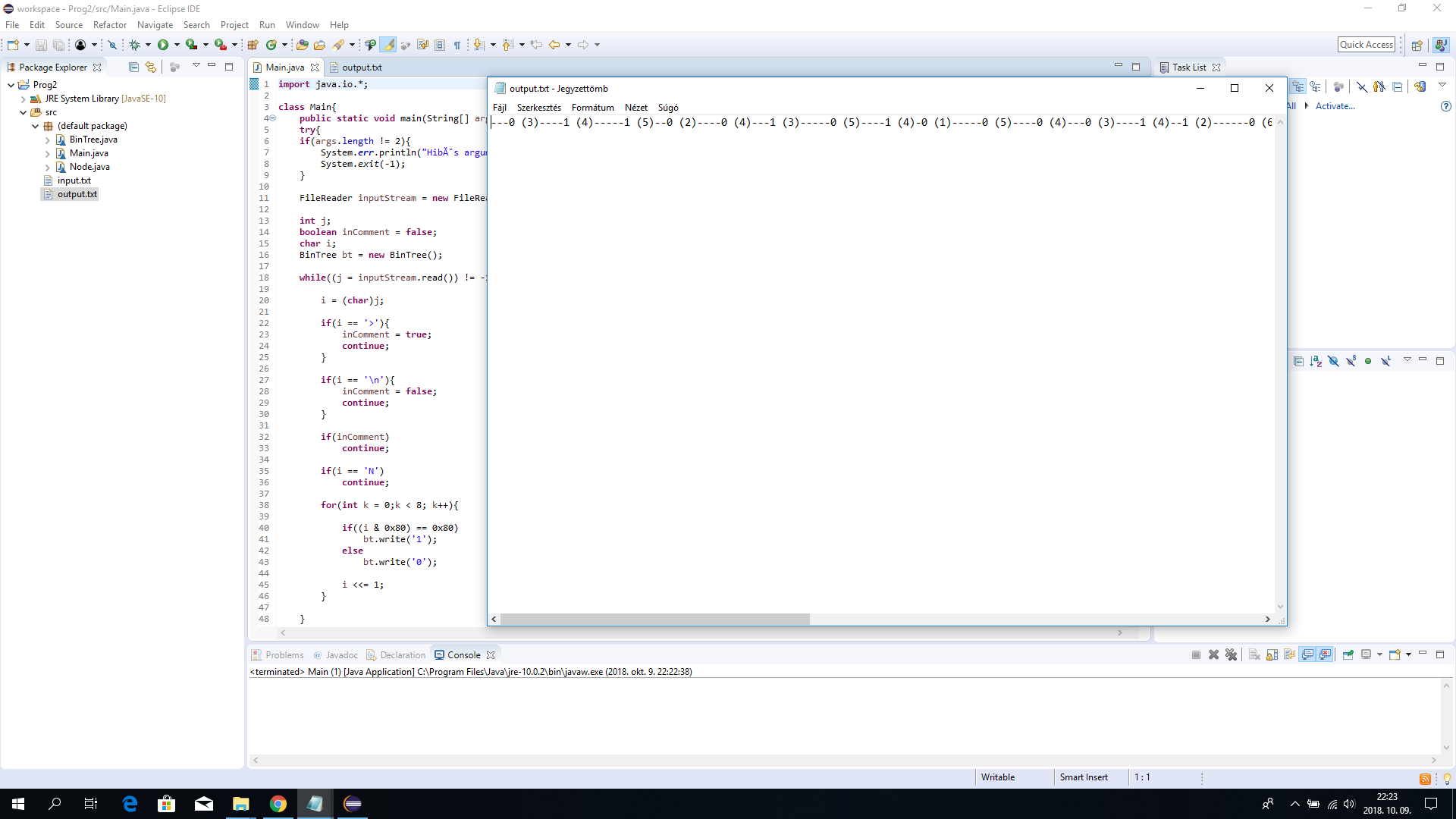
A feladat megoldásához tutorként Bátfai Norbert tanár úr forráskódját vettem igénybe.

igyekeztem a feladathoz hűen rámutatni a pointerekre illetve a referenciák.

A feladatot értelmeztem és sikeresen fordítottam, futtattam.

Source Code:

<https://github.com/megvagyhadnagy01/MagasProg2/tree/master/1_gyak/Homokozo>



------------------------------------------------------------------------------------------------------------------------->

**“Gagyi”**

A Gagyi feladat megoldásánál rámutat a különbségre hogy a program hogy reagál arra ha -128nál kisebb számot adunk meg

és a while ciklus végtelen lesz.JDK forrásai szintjén:

vegyük a -129 esetet, ami fagy, azaz végtelen ciklus, ekkor

i-ben a -129 van

public static Integer valueOf(int i) {

amivel ez az if

if (i >= IntegerCache.low && i <= IntegerCache.high)

nem teljesül (ellenőrizd az Integer.java forrásban!)

return IntegerCache.cache[i + (-IntegerCache.low)];

tehát ez fog lefutni mindkét esetben, amikor a Gagyi progiban csináljuk a két Integert

return new Integer(i);

}

A válasz innen már trivi: a két integernél két új objektum jön létre, máshol a tárban más "mem.címmel" persze,

így a hasonlításuk hamis lesz, hiszen különböző objectek, különböző "címmel".

A másik esetben (a -128 esetén) pedig ugyanarra a számra ugyanazt az objectet adja,

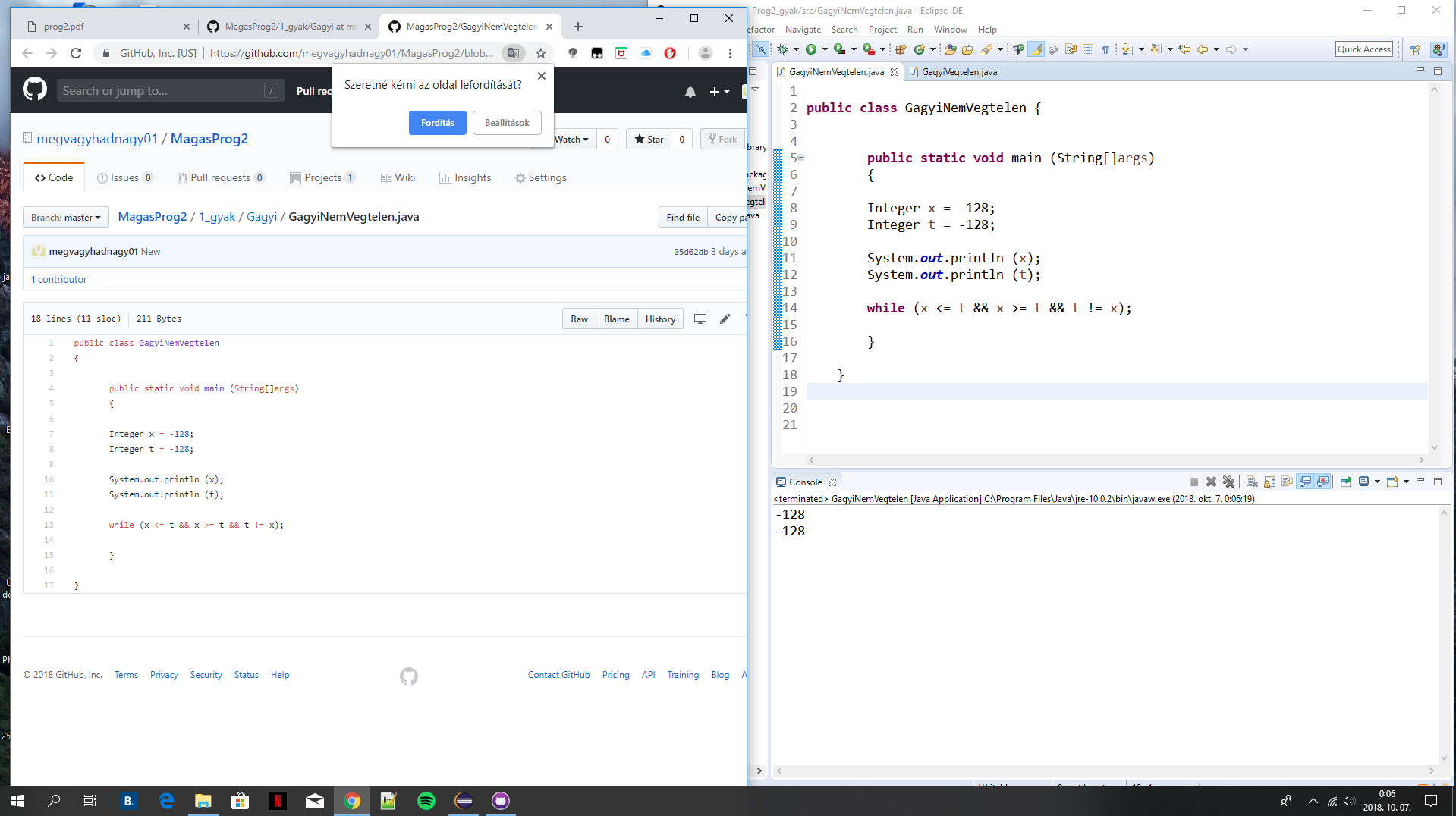
hiszen csak kiveszi a kész konzerv integert a poolból, így nyilván ez a két cím meg == lesz.

Source code:

<https://github.com/megvagyhadnagy01/MagasProg2/tree/master/1_gyak/Gagyi>

Futtatás:

<https://github.com/megvagyhadnagy01/MagasProg2/blob/master/1_gyak/Gagyi/GagyiNemVegtelen.PNG>



------------------------------------------------------------------------------------------------------------------------->

**Yoda:**

A Yoda feladatnál egy saját kódot írtam ami sikeresen fordult és fut.Létre hoztam egy “yod” string tiposú változot amit null értékel pédányositotam., majd azt egy if fügvényel megnéztem ,hogy megegyezik-e az értéke a yodal .Ez utén igyekeztem megfogni a lényeget a feladatnál try illetve catch függvények és

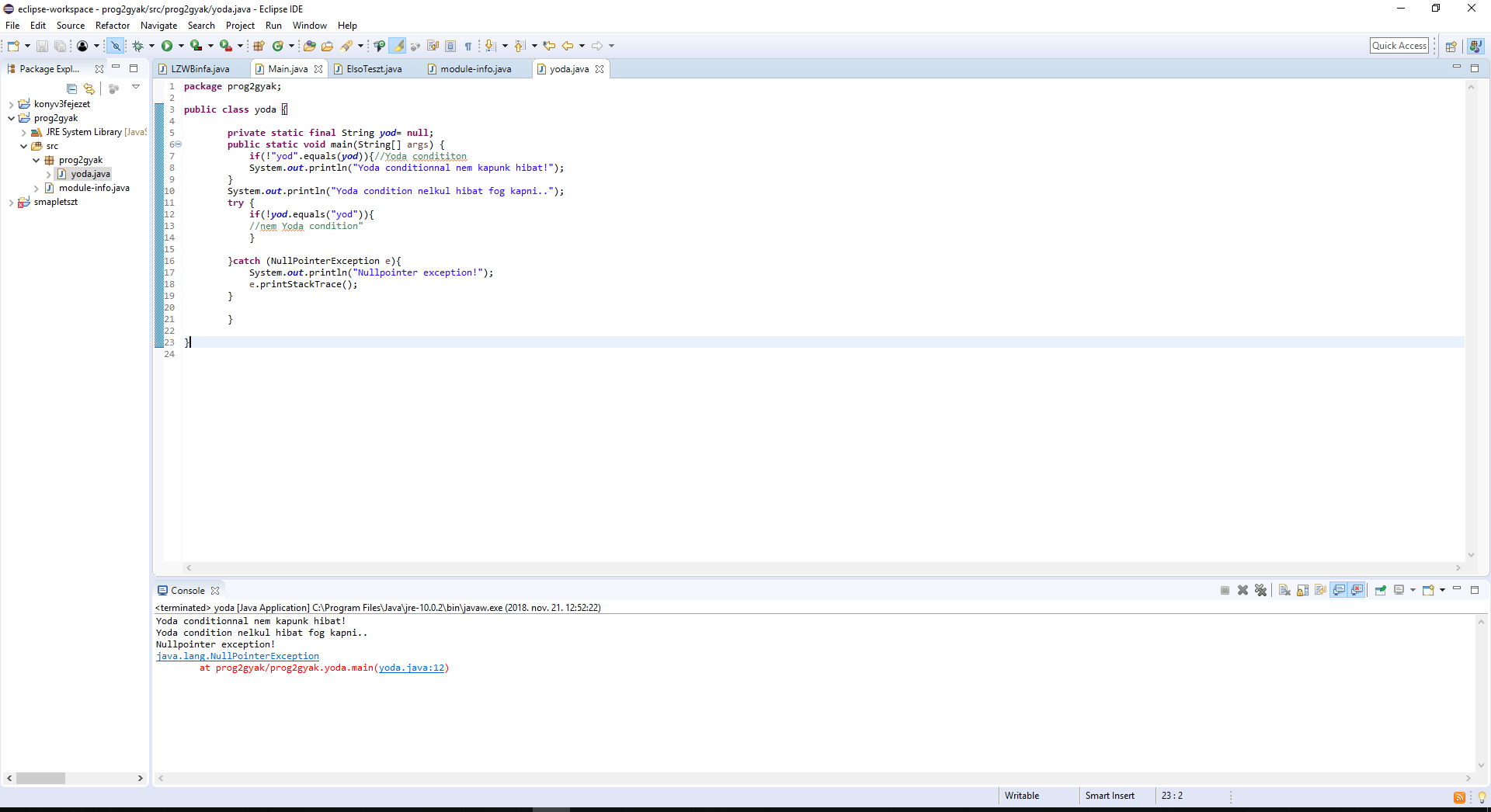
hasonlo képpen NUllPointerException kivétel kezelésel.

Source code:

<https://github.com/megvagyhadnagy01/MagasProg2/tree/master/1_gyak/Yoda>

Futtatás:

<https://github.com/megvagyhadnagy01/MagasProg2/blob/master/1_gyak/Yoda/Yoda.PNG>



------------------------------------------------------------------------------------------------------------------------->

**Kódolás from scratch :**

﻿A feladat megoldásához tutorként Bátfai Norbert tanár úr forráskódját vettem igénybe.

Illetve a megadott angol nyelvű forrást igyekeztem értelmezni és az alapján is felfogni a feladatot

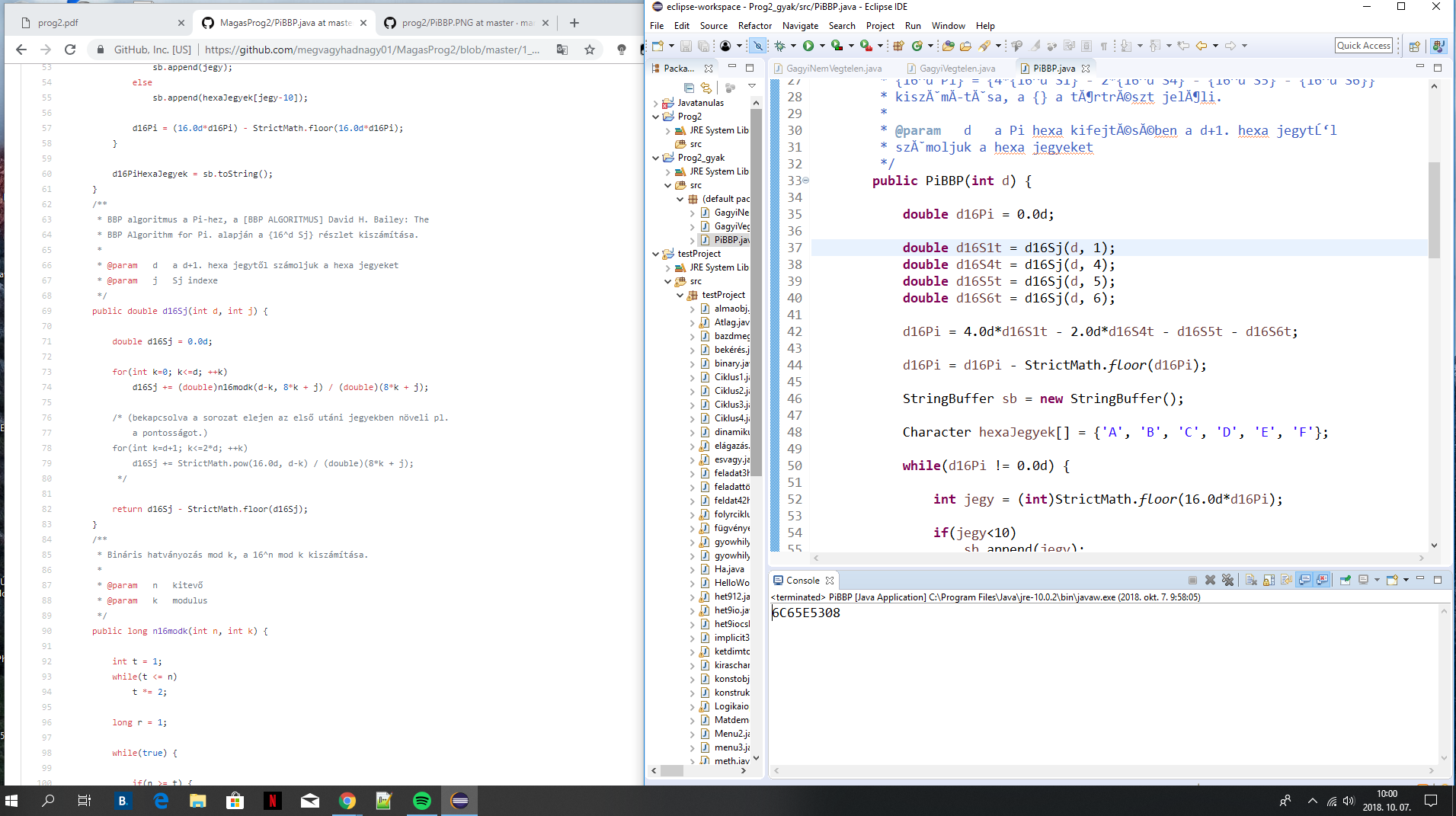
A feladatot értelmeztem és sikeresen fordítottam, futtattam.

Source code:

<https://github.com/megvagyhadnagy01/MagasProg2/tree/master/1_gyak/PiBBP>

Futtatás:

<https://github.com/megvagyhadnagy01/MagasProg2/blob/master/1_gyak/PiBBP/PiBBP.PNG>

****

2.hét

**Anti OO:**

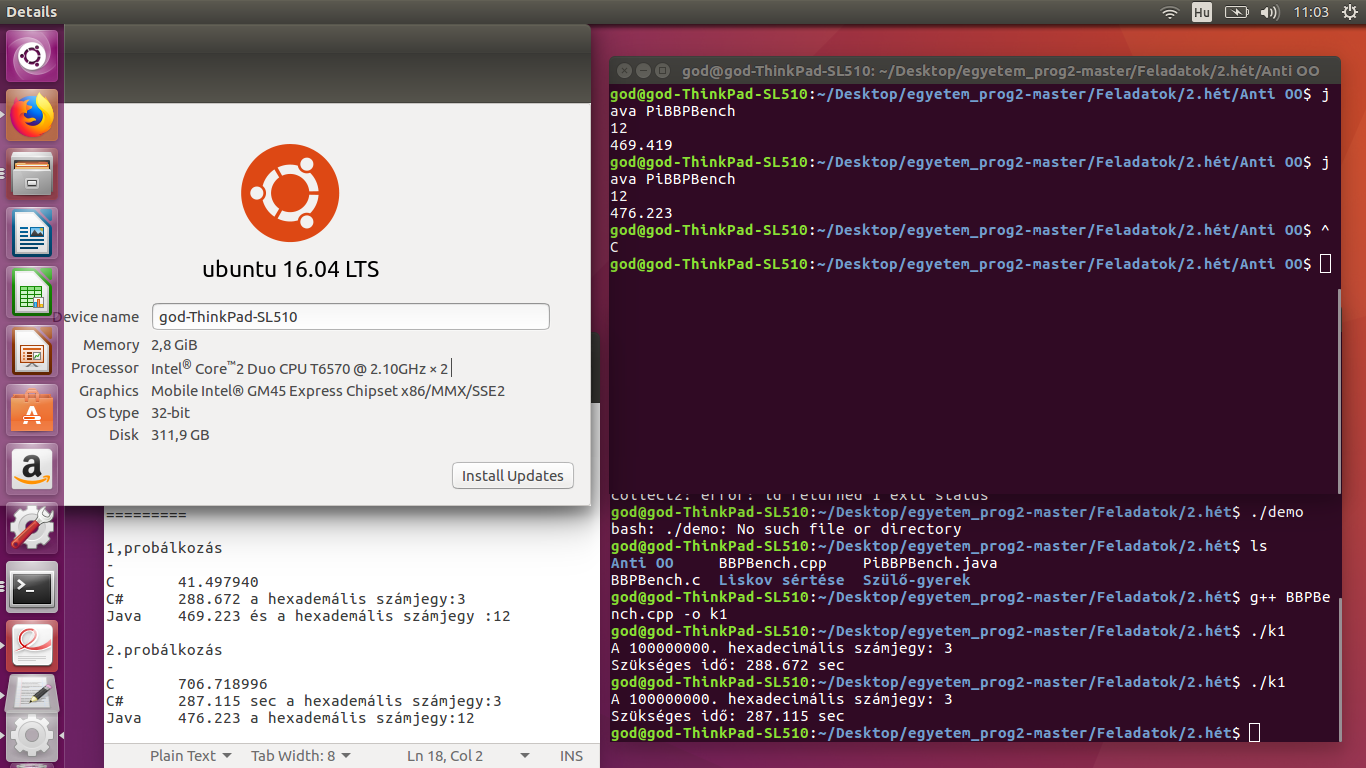
Az algoritmust tutor segítségével sikerult megoldani.Sikeresen értelmeztem és futtatom a programot.

Source code:

<https://github.com/megvagyhadnagy01/MagasProg2/tree/master/2_gyak/Anti%20OO>

Futtatás:

<https://github.com/megvagyhadnagy01/MagasProg2/blob/master/2_gyak/Anti%20OO/Screenshot%20from%202018-10-09%2011-03-24.png>

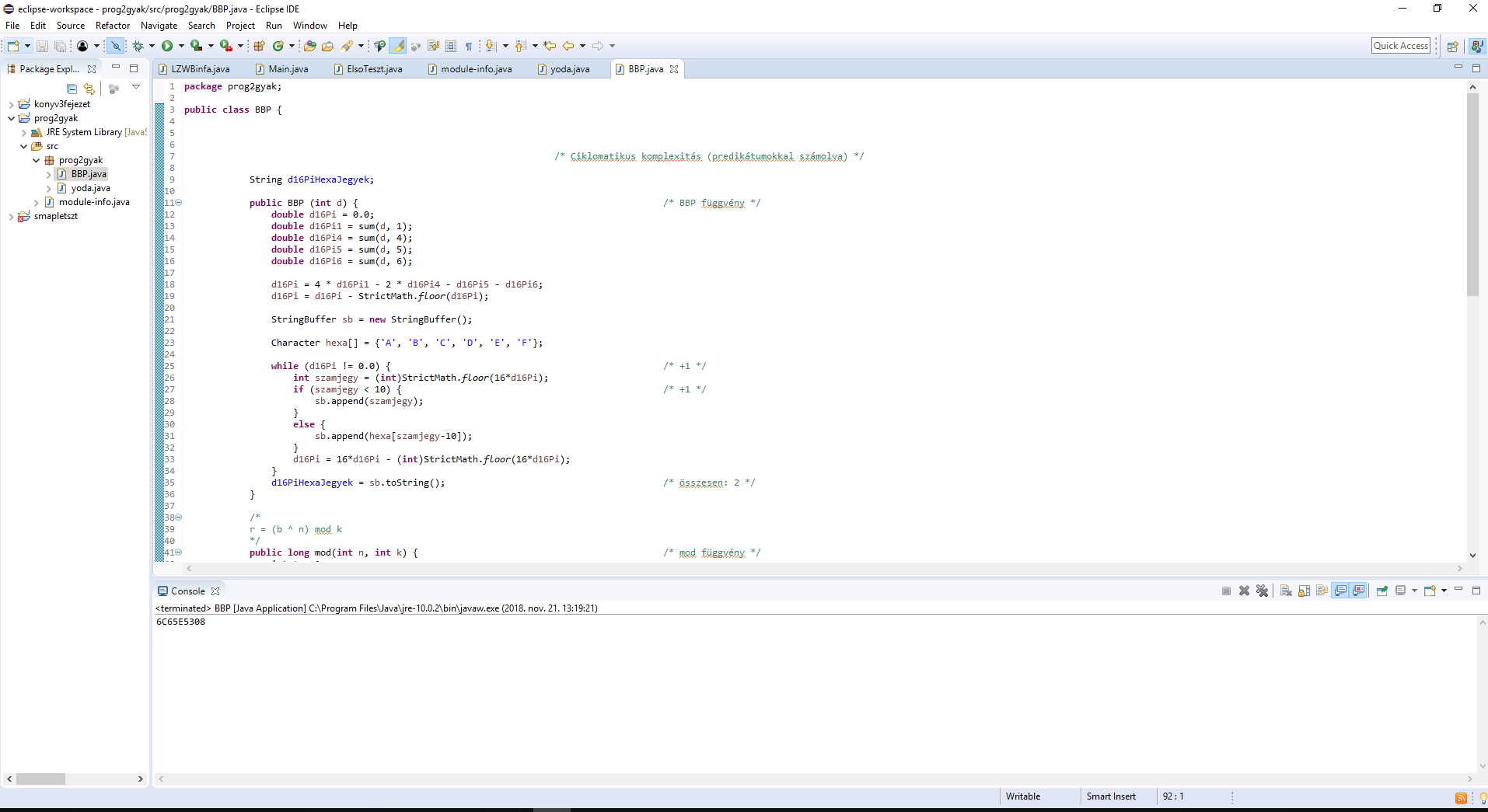


------------------------------------------------------------------------------------------------------------------------->

**Ciklomatikus komplexitás:**

Source code:

<https://github.com/megvagyhadnagy01/MagasProg2/tree/master/2_gyak/Ciklo>



------------------------------------------------------------------------------------------------------------------------->

**Szülő-gyerek:**

Minden objektum referencia. Mindig dinamikus a kötés. De ezzel nem küldhetjük a Gyerek által hozott új üzeneteket.Csak akkor van dinamikus kötés, ha a viselkedés virtuálisra (virtual kulcsszó az ősben) van deklarálva. Ugyanúgy igaz, hogy ösosztály referencián vagy pointeren keresztül, csak az ős üzenetei küldhetőek.

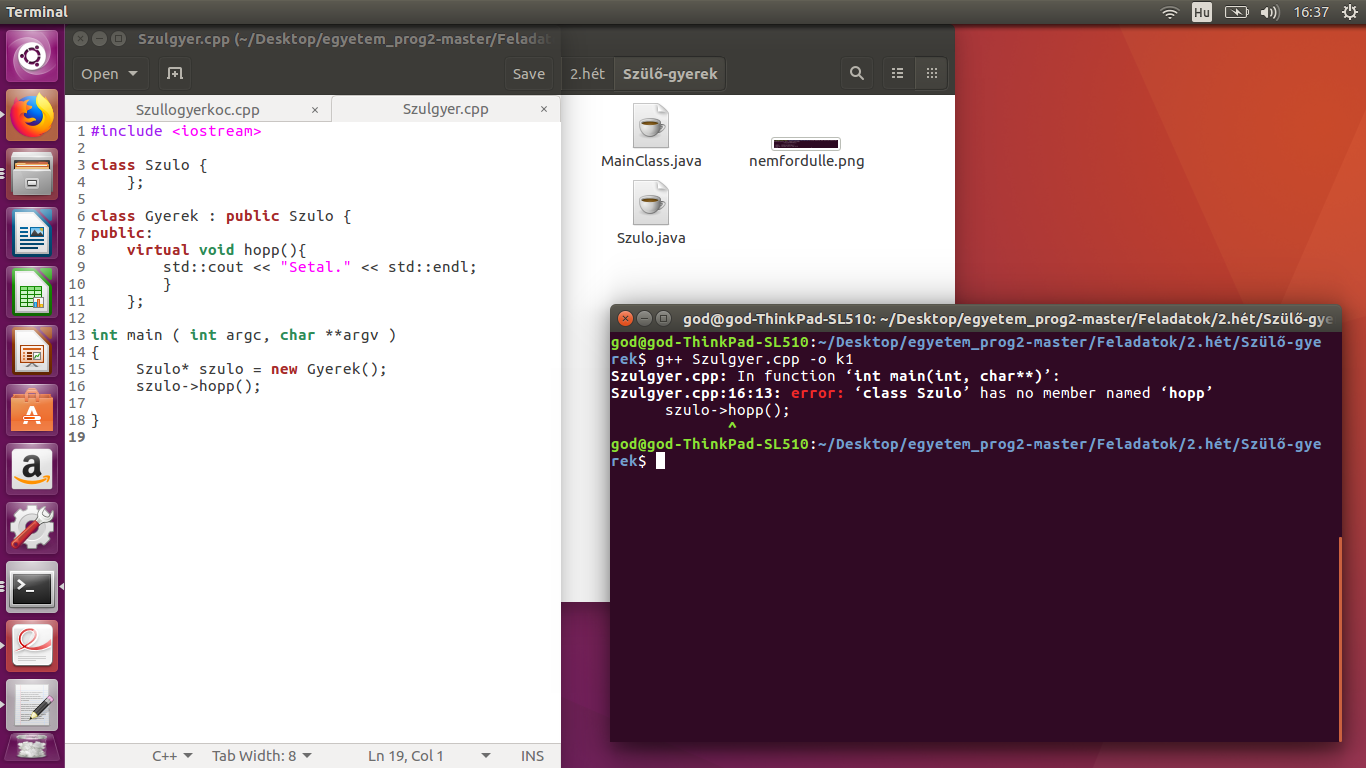
Source code:

<https://github.com/megvagyhadnagy01/MagasProg2/tree/master/2_gyak/SzuloGyerek>

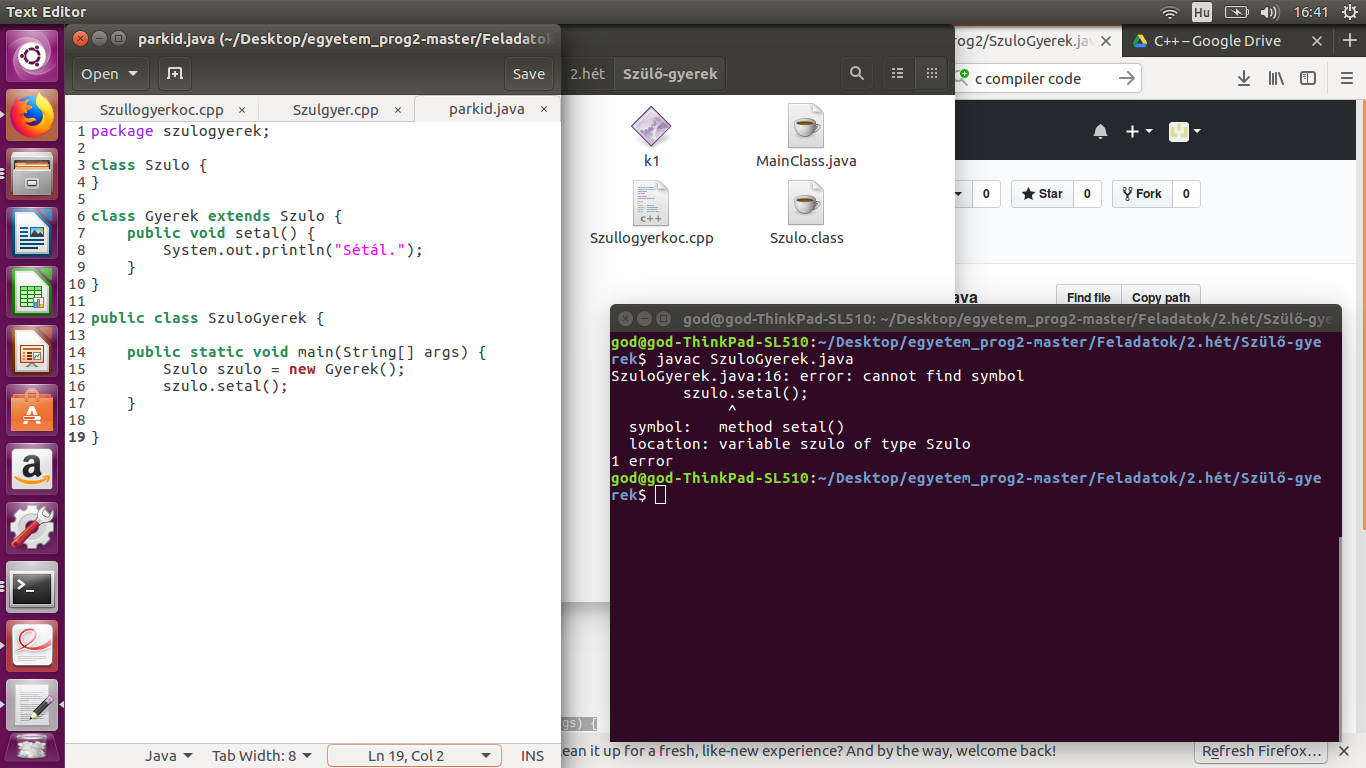
Futtatás:

<https://github.com/megvagyhadnagy01/MagasProg2/blob/master/2_gyak/SzuloGyerek/Screenshot%20from%202018-10-09%2016-41-16.png>

c++:



Java:



------------------------------------------------------------------------------------------------------------------------->

**Liskov:**

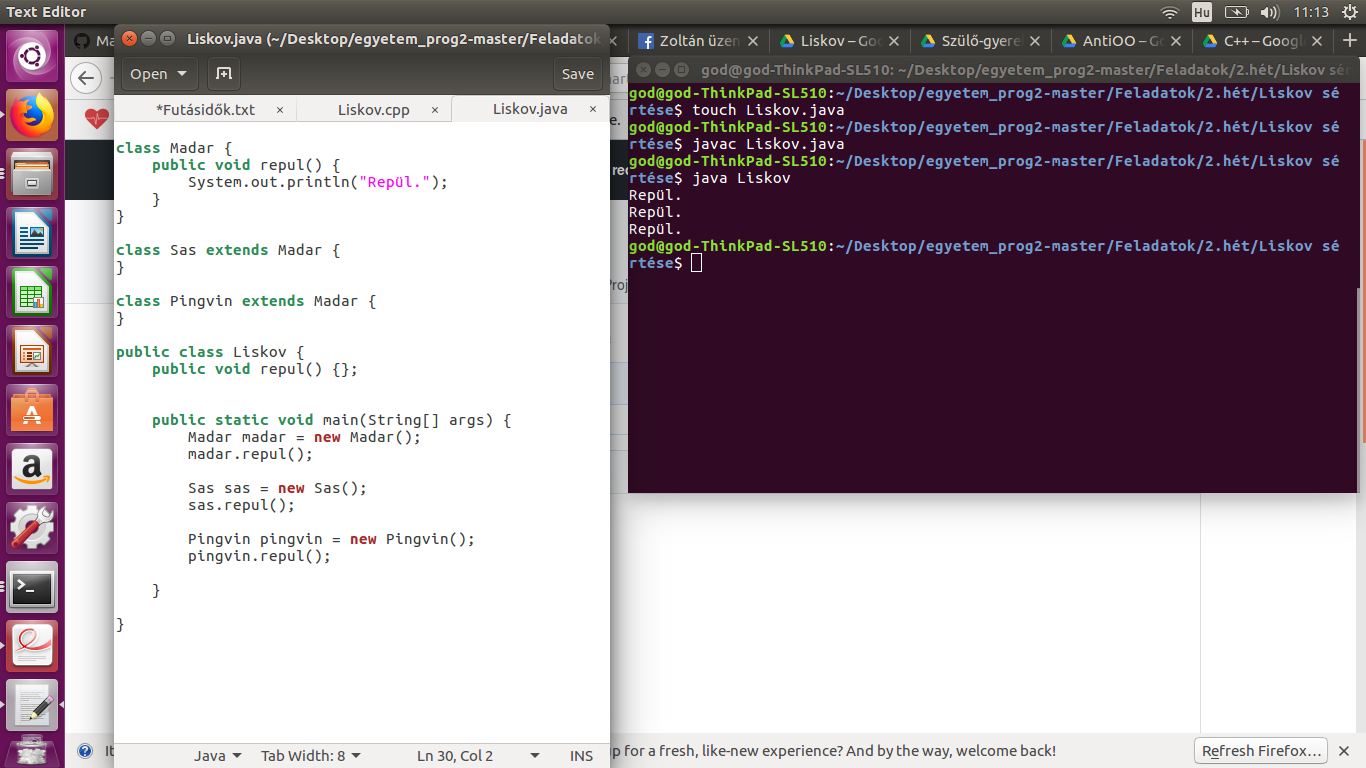
Az elöadás foliája alapján igyekeztem megirni a kodot .melyben (Szulo\*szulo = new Gyerek();) öröklődés következtében melyben az ösökkel működő függvény működjön a gyermekkel is.

Source code:

<https://github.com/megvagyhadnagy01/MagasProg2/tree/master/2_gyak/liskov>

Futtatás:

<https://github.com/megvagyhadnagy01/MagasProg2/blob/master/2_gyak/liskov/Screenshot%20from%202018-10-09%2011-13-12.png>



------------------------------------------------------------------------------------------------------------------------->

3.hét

**BPMN:**

A feladatot a bpmn.io oldalon hajtottam végre ahol egy pizzéria működéséről készítettem diagramot.

Source code:

<https://github.com/megvagyhadnagy01/MagasProg2/tree/master/3_gyak/BPMN>

Futtatás:

<https://github.com/megvagyhadnagy01/MagasProg2/blob/master/3_gyak/BPMN/diagram.svg>

------------------------------------------------------------------------------------------------------------------------->

**Forward engineering UML osztálydiagram :**

Source code:

<https://github.com/megvagyhadnagy01/MagasProg2/tree/master/3_gyak/Foward>

Futtatás:

<https://github.com/megvagyhadnagy01/MagasProg2/blob/master/3_gyak/Foward/Forwarduml.png>

------------------------------------------------------------------------------------------------------------------------->

**Reverse engineering UML osztálydiagram**

Source code:

<https://github.com/megvagyhadnagy01/MagasProg2/tree/master/3_gyak/Reverse%20engineering%20UMl>

Futtatás:

<https://github.com/megvagyhadnagy01/MagasProg2/blob/master/3_gyak/Reverse%20engineering%20UMl/eredm%C3%A9ny.png>

------------------------------------------------------------------------------------------------------------------------->

**“Egy esettan”** :

Igyekeztem értelmezni a megadott feladatokat és meglévő kódokat.Lefuttattam.

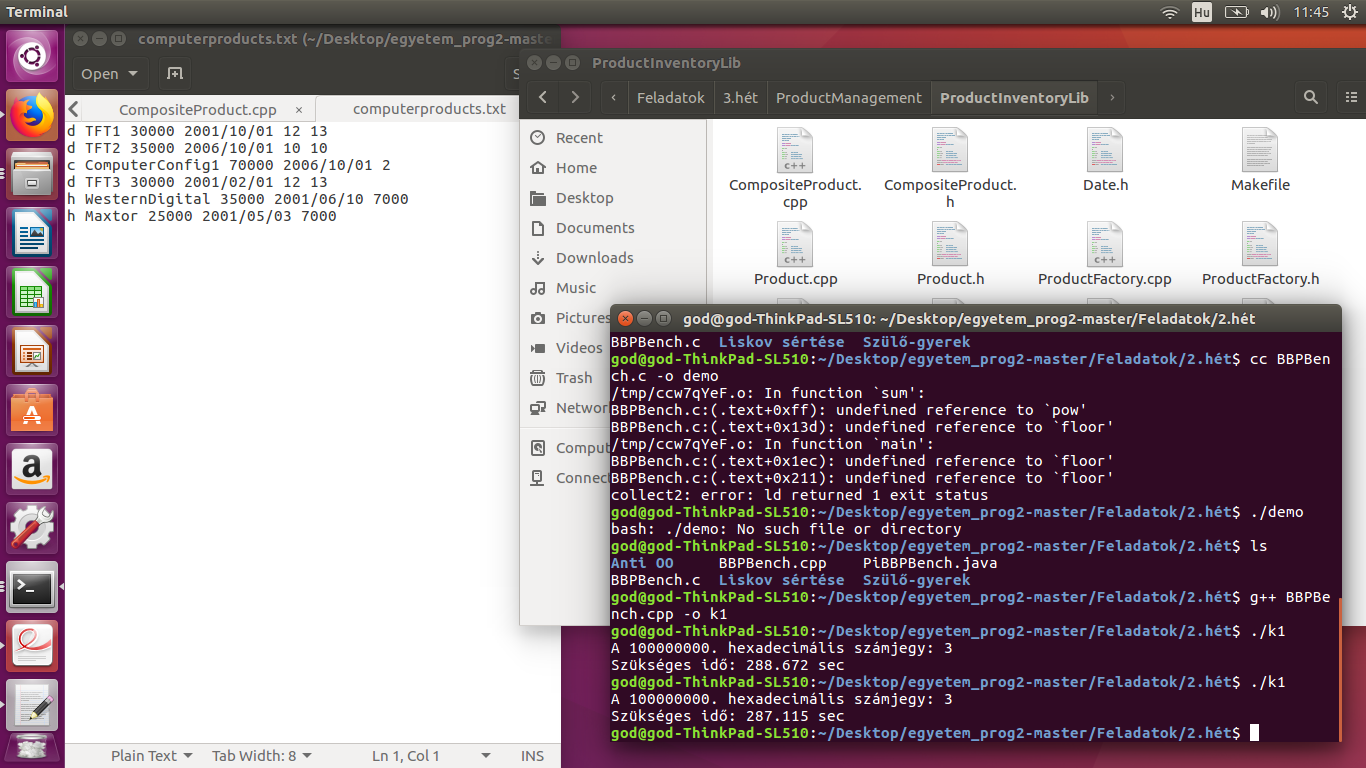
A BME-s C++ tankönyv 14. fejezetét (427-444 elmélet, 445-469 az esettan) dolgoztam fel.

Source code:

<https://github.com/megvagyhadnagy01/MagasProg2/tree/master/3_gyak/ProductManagement>

Futtatás:

<https://github.com/megvagyhadnagy01/MagasProg2/blob/master/3_gyak/ProductManagement/Screenshot%20from%202018-10-09%2011-45-36.png>



------------------------------------------------------------------------------------------------------------------------->

4.hét

**Encoding :**

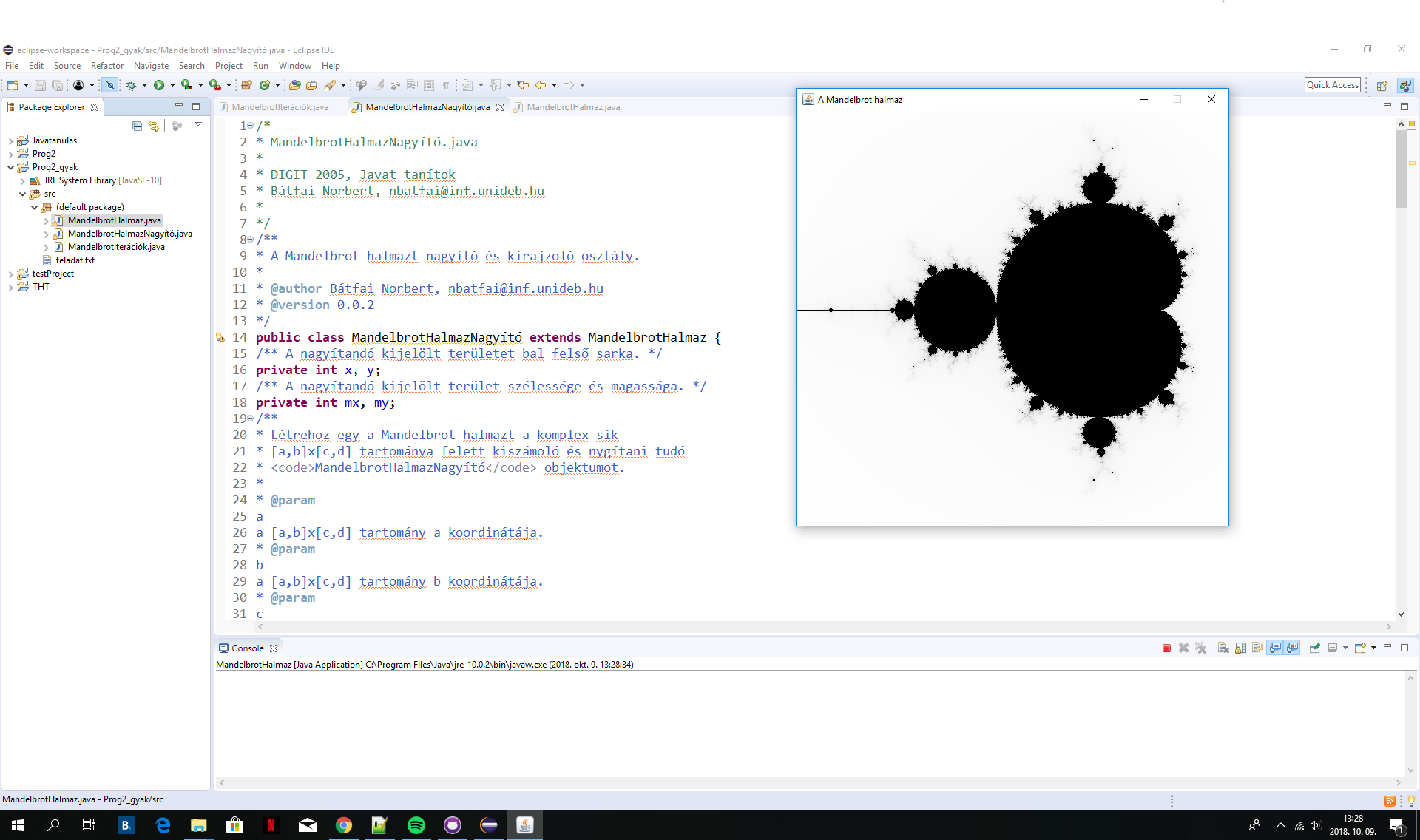
A netbeans alap karakterkódolás Utf-8 a file ok pedig olyan karakterkódolást tartalmaznak mai nincs beállítva ezért át kell állítani windows-1252-re.Sikeresen lefordult a program.

Source code:

<https://github.com/megvagyhadnagy01/MagasProg2/tree/master/4.gyakorlat/Encoding>

Futtatás:

<https://github.com/megvagyhadnagy01/MagasProg2/blob/master/4.gyakorlat/Encoding/MandelBrot.PNG>



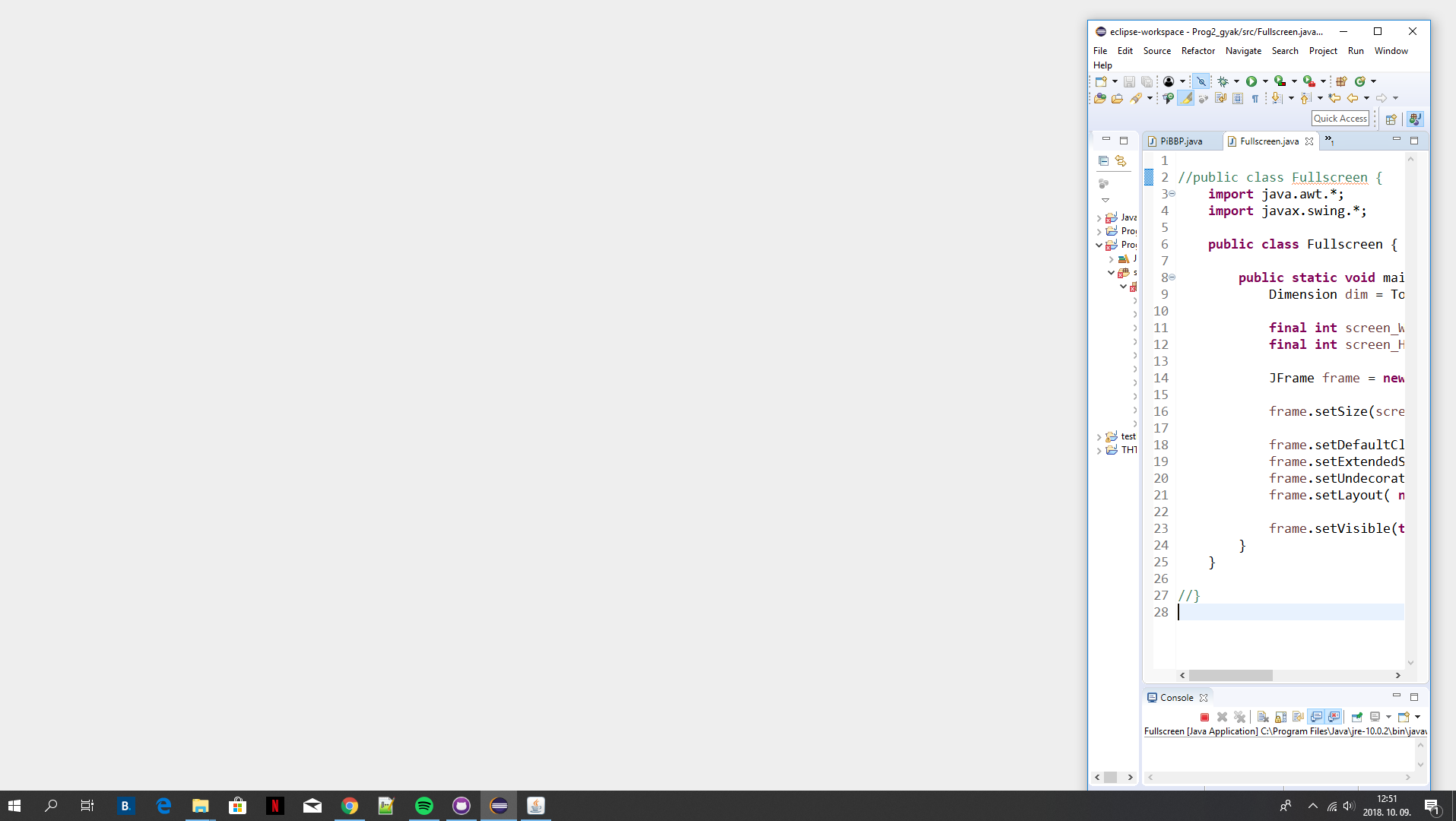
------------------------------------------------------------------------------------------------------------------------->

**Full screen:**

A feladat lényege java grafikus képességét szemlélteti.Frame függvény használatával karöltve.

Source code:

<https://github.com/megvagyhadnagy01/MagasProg2/tree/master/4.gyakorlat/Fullscreen>

Futtatás:<https://github.com/megvagyhadnagy01/MagasProg2/blob/master/4.gyakorlat/Fullscreen/Fullscrenn.PNG>

------------------------------------------------------------------------------------------------------------------------->

**OOCWC lexer:**

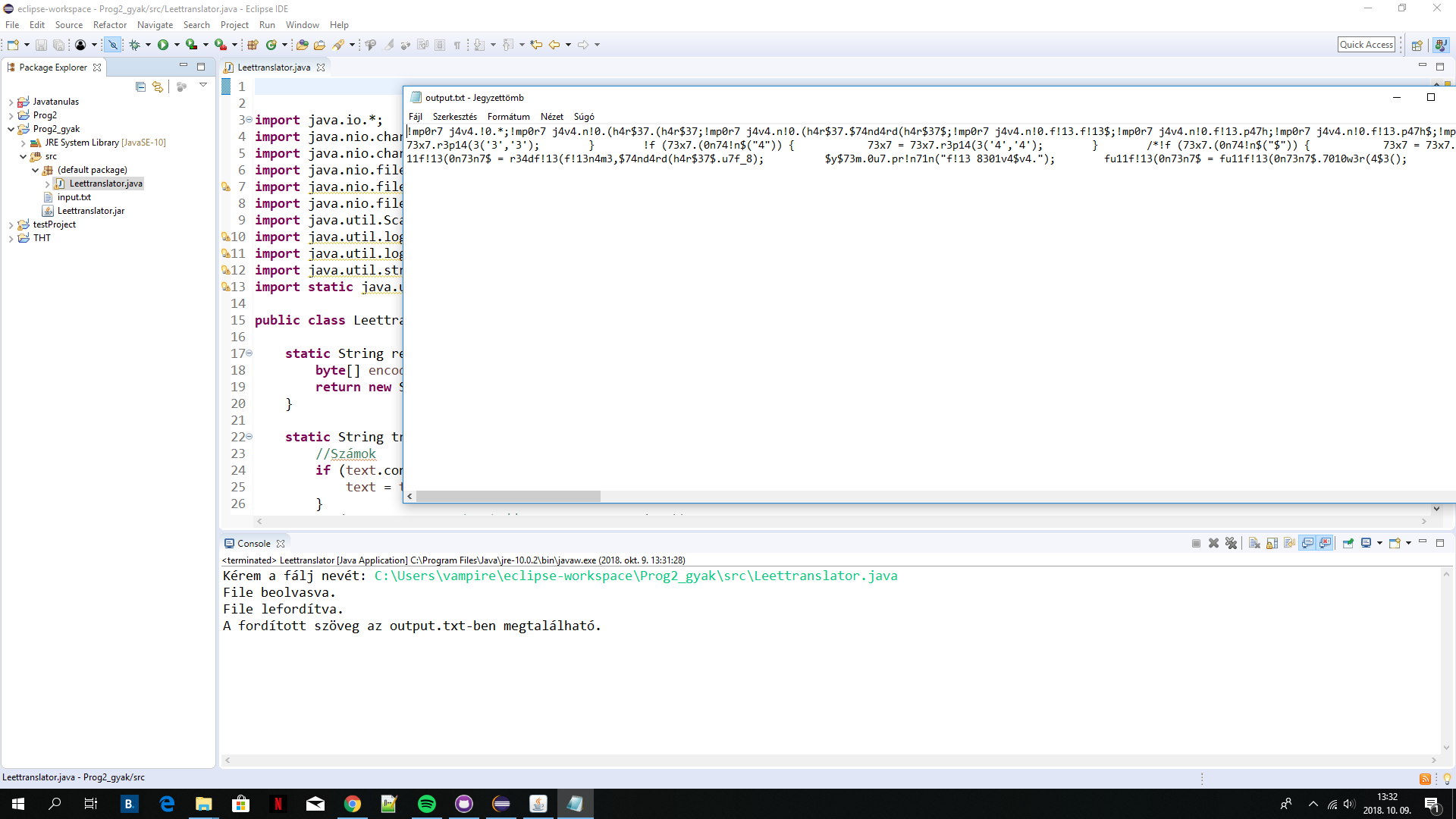
A carlexer.ll fájl elején rövidítések vannak definiálva, amelyek értékei vagy konkrét karaktersorozat ként, vagy reguláris kifejezésként vannak megadva.A lexer ezeket a rövidítéseket használja fel mintaként, ha pedig a megadott minták valamelyikével egyezést talál, akkor annak a szövegét megfelelő formátumúvá alakítja át,és ezt memóriacímen tárolja.

Source code:

<https://github.com/megvagyhadnagy01/MagasProg2/tree/master/4.gyakorlat/Leettranslator>

Futtatás:

<https://github.com/megvagyhadnagy01/MagasProg2/blob/master/4.gyakorlat/Leettranslator/Lettranslator.PNG>



------------------------------------------------------------------------------------------------------------------------->

5.hét

**JDK osztályok:**

Ezt terjedelmesebben az összefoglalóban igyekeztem bemutatni.De léynegében Boost C++ programot kell hazsnálni és ott egyik részlét beimpelmentéálni

.hogy ki tudjuk listázni a JDK osztályokat.

Source Code:

<https://github.com/megvagyhadnagy01/MagasProg2/tree/master/5.gyakorlat/JDK%20oszt%C3%A1lyok>

Futtás:

------------------------------------------------------------------------------------------------------------------------->

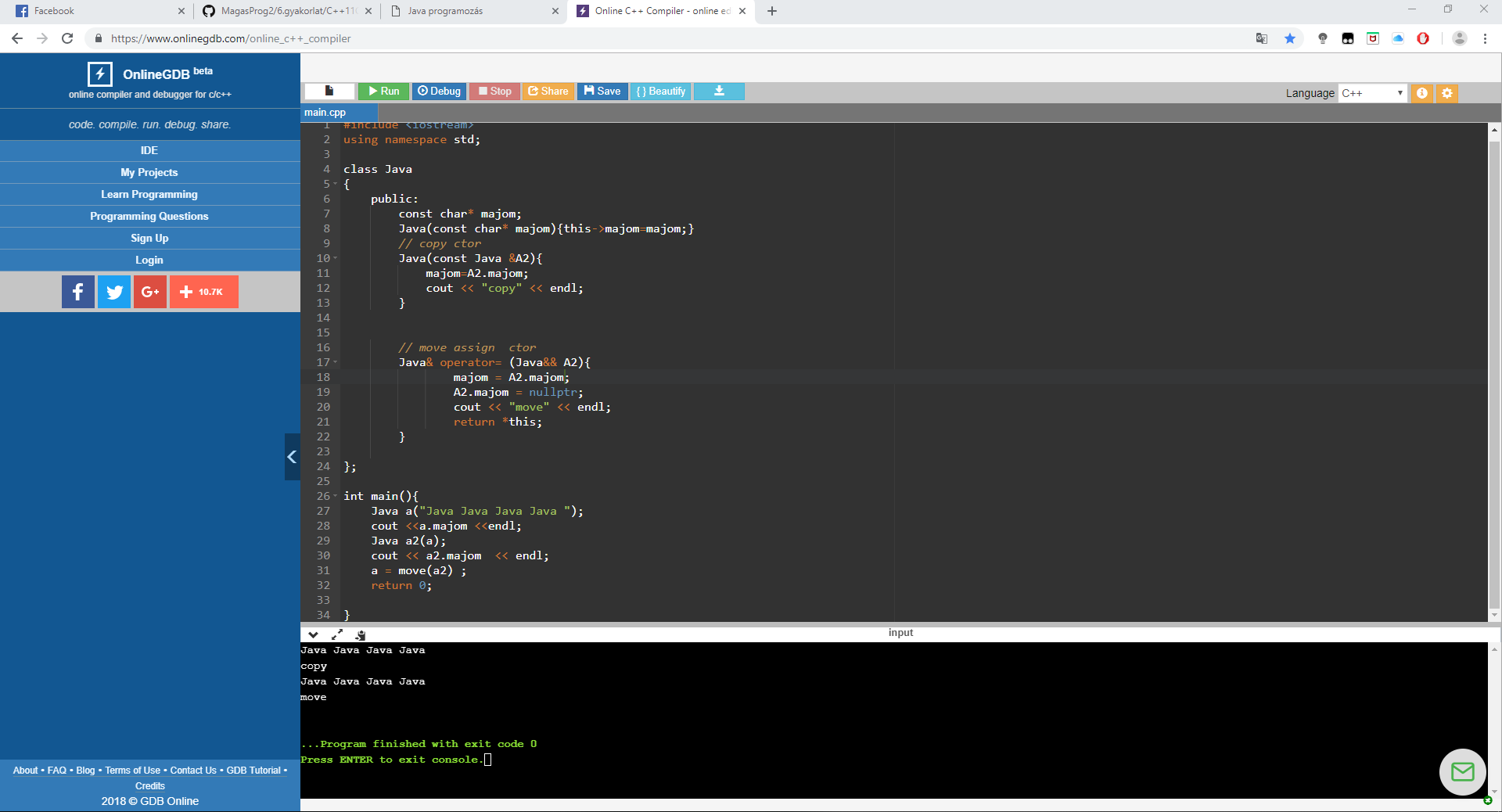
**Másoló-mozgató szemantika :**

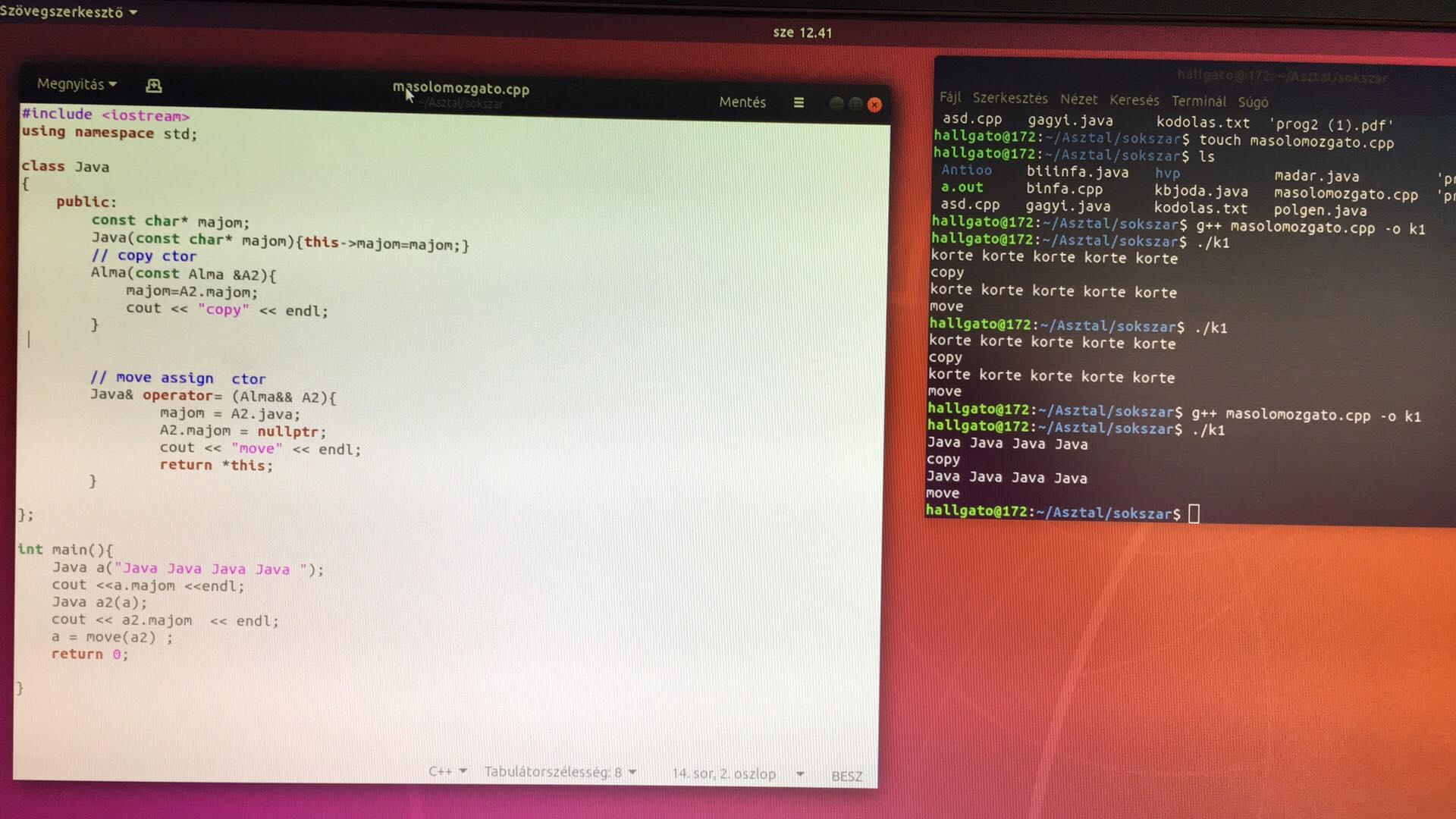
Source code:

<https://github.com/megvagyhadnagy01/MagasProg2/tree/master/5.gyakorlat/M%C3%A1solo%26Mozgato>

Futtatás:

<https://github.com/megvagyhadnagy01/MagasProg2/blob/master/5.gyakorlat/M%C3%A1solo%26Mozgato/masolo.PNG>





------------------------------------------------------------------------------------------------------------------------->

**Hibásan implementált RSA törése:**

Itt lényegében hibásat /tört kódból kell implementálni titkosított sz9veget.

Az elöadás foliája alapján igyekeztem létrehozni akodomat meklyben array ben tároltam a tisztaszio veget

melyet a length függvény összeszámoltam és azt bevittem egy for függvény a java.math.BitEngert

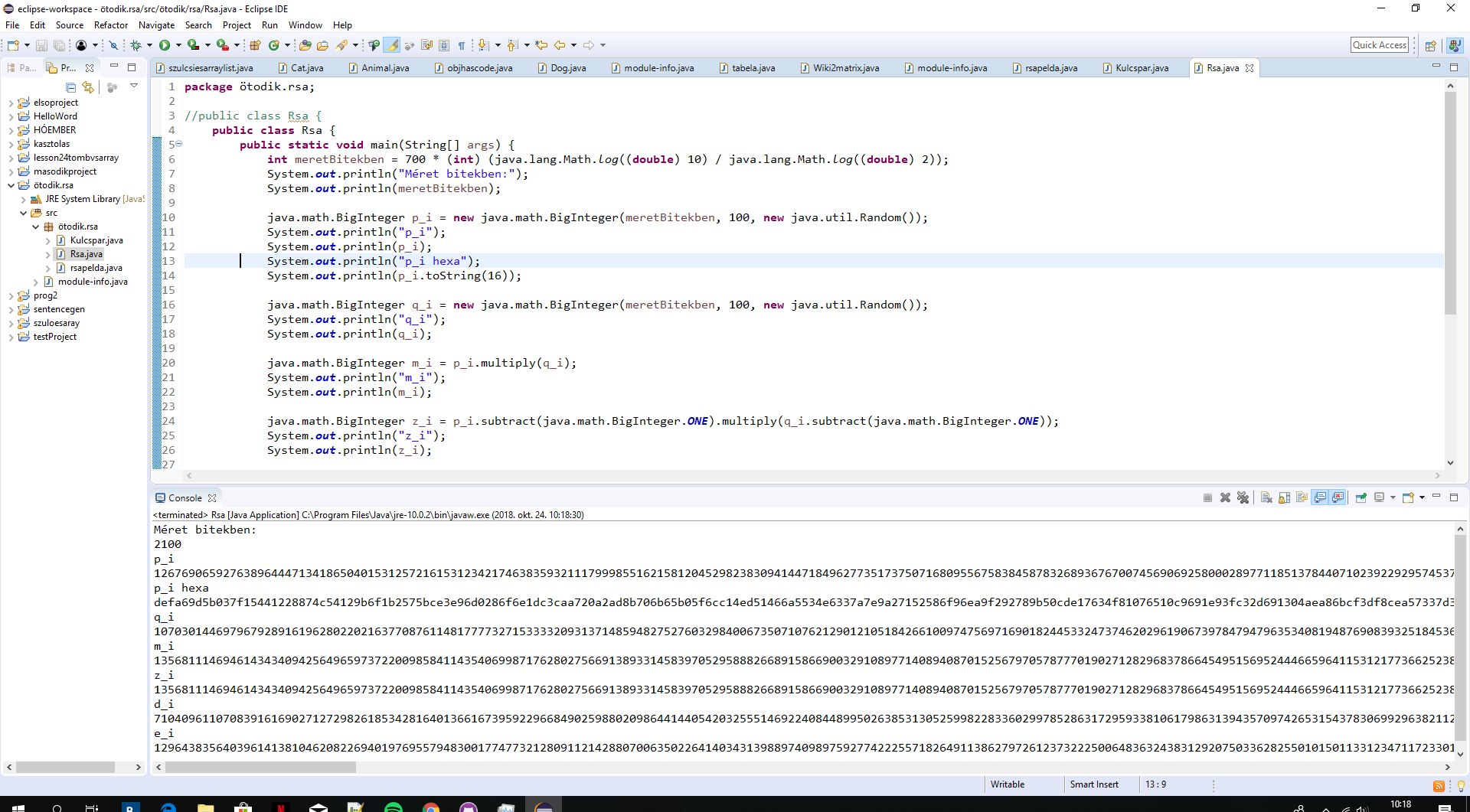
Majd a kódolásnál pedig for függvénnyel teszteltem a java.modPow használtam.

Source code:

<https://github.com/megvagyhadnagy01/MagasProg2/tree/master/5.gyakorlat/RSA>

Futtatás:

<https://github.com/megvagyhadnagy01/MagasProg2/blob/master/5.gyakorlat/RSA/Rsa.PNG>



------------------------------------------------------------------------------------------------------------------------->

**Változó argumentumszámú ctor :**

Source code:

<https://github.com/megvagyhadnagy01/MagasProg2/tree/master/5.gyakorlat/V%C3%A1ltozoargamentumsz%C3%A1m>

------------------------------------------------------------------------------------------------------------------------->

Összefoglaló:

<https://github.com/megvagyhadnagy01/MagasProg2/blob/master/5.gyakorlat/Essz%C3%A9cske.txt>

------------------------------------------------------------------------------------------------------------------------->

6.hét