Eötvös Loránd Tudományegyetem

Informatikai Kar

Programozási nyelvek és Fordítóprogramok Tanszék

Java bytecode interpreter Javában

Témavezető:
Kozsik Tamás Dr.
egyetemi docens

Szerző:
Balázs Zoltán
programtervező informatikus BSc

Budapest, 2023

EÖTVÖS LORÁND TUDOMÁNYEGYETEM
INFORMATIKAI KAR

SZAKDOLGOZAT TÉMABEJELENTŐ

Hallgató adatai:
Név: Balázs Zoltán
Neptun kód: HV56L5

Képzési adatok:
Szak: programtervező informatikus, alapképzés (BA/BSc/BProf)
Tagozat : Nappali

Belső témavezetővel rendelkezem

Témavezető neve: *Kozsik Tamás Dr.*
munkahelyének neve, tanszéke: ***ELTE IK, Programozási nyelvek és Fordítóprogramok Tanszék***
munkahelyének címe: ***1117, Budapest, Pázmány Péter sétány 1/C.***
beosztás és iskolai végzettsége: ***egyetemi docens, programtervező matematikus***

A szakdolgozat címe: Java bytecode interpreter Javában

A szakdolgozat témája:
(A témavezetővel konzultálva adja meg 1/2 - 1 oldal terjedelemben szakdolgozat témájának leírását)

A Java nyelvben írt programok fordításuk során nem közvetlenül gépi kódra fordulnak, hanem egy hardver-független nyelvre, amit bytecode-nak neveznek.
Ezt a bytecode-ot az esetek többségében a JVM (Java Virtual Machine) interpreter-e hajtva végre, vagy futási időben fordul le a fordító gép hardverének gépi kódjára.

A szakdolgozat célja egy olyan Java bytecode interpreter fejlesztése, amely képes már előre, valamilyen Java fordító által, elkészített bytecode-ot interpreter-álni, ezt sikeresen (és helyesen) lefuttatni.
A fejlesztett interpreter-nek képesnek kell lennie az ELTE Programtervező Informatikus BSc szakán, különböző, Java-t használó tárgyakon (Programozási nyelvek, Konkurens programozás) elkészített beadandók és házi feladatok generált bytecode-ját interpreter-álni, ezeket helyesen futtatni.

Budapest, 2022. 11. 24.

Tartalomjegyzék

1. fejezet

Bevezetés

A Java nyelvben írt programok fordításukat követően nem egy közletlen futtatható állományra (gépi kódra) fordulnak (a fordítást általában a beépített `javac` program végzi el), hanem egy köztes nyelvre, bytecode-ra, amelyet aztán különböző programokkal az adott architektúrán interpretáljuk. Legtöbb esetben az interpretálást a JVM (Java Virtual Machine) interpretere hajtja végre (ez a beépített `java` program).

A szakdolgozat célja egy kiegészítő program (fantázianevén *Jabyinja* – ***Ja****va* ***by****tecode* ***in****terpreter in* ***Ja****va*) írása, amely ugyan hagyatkozik a `javac` és `java` programokra (az előbbire a fordítás, az utóbbira a futtatás miatt), de a tényleges futtatást a különböző bytecode instrukciók implementálásval végzi el.

A program nincsen Java kód interpretálásához kötve, a Java bytecode a neve ellenére más programozási nyelveknek is az alapja (ezek közül az ismertebbek: Kotlin, Clojure), viszont a tesztelés csak Java kódból generált bytecodera tér ki, ugyanis a szakdolgozat céljaként az ELTE Programtervező Informatikus BSc szakán elkészített Java programok fordításának interpretálását tűztem ki.

A programnak szükséges értelmeznie kell egy adott Classfájlt (többet is ha egy külön fájlra is hivatkozunk), helyesen beolvasnia a benne lévő adatokat, majd a belépési (*main*) metódust lefuttatnia. A program erősen alapszik a Java nyelvbe beépített reflekcióra, ezen felül saját stack implementálása is szükséges. Mivel a Java nyelvre épül a program, ezért saját heap megírására nincsen szükség, ez automatikusan kezelve lesz.

2. fejezet

Felhasználói dokumentáció

Lorem ipsum dolor sit amet $\mathbb{N}$, consectetur adipiscing elit. Duis nibh leo, dapibus in elementum nec, aliquet id sem. Suspendisse potenti. Nullam sit amet consectetur nibh. Donec scelerisque varius turpis at tincidunt. Cras a diam in mauris viverra vehicula. Vivamus mi odio, fermentum vel arcu efficitur, lacinia viverra nibh. Aliquam aliquam ante mi, vel pretium arcu dapibus eu. Nulla finibus ante vel arcu tincidunt, ut consectetur ligula finibus. Mauris mollis lectus sed ipsum bibendum, ac ultrices erat dictum. Suspendisse faucibus euismod lacinia $\mathbb{Z}$.

2.1. Felsorolások

Etiam vel odio ante. Etiam pulvinar nibh quis massa auctor congue. Pellentesque quis odio vitae sapien molestie vestibulum sit amet et quam. Pellentesque vel dui eget enim hendrerit finibus at sit amet libero. Quisque sollicitudin ultrices enim, nec porta magna imperdiet vitae. Cras condimentum nunc dui, eget molestie nunc accumsan vel.

- Fusce in aliquet neque, in pretium sem.
- Donec tincidunt tellus id lectus pretium fringilla.
- Nunc faucibus, erat pretium tempus tempor, tortor mi fringilla neque, ac congue ex dui vitae mauris.

Donec dapibus sodales ante, at scelerisque nunc laoreet sit amet. Mauris porttitor tincidunt neque, vel ullamcorper neque pulvinar et. Integer eu lorem euismod,

faucibus lectus sed, accumsan felis. Nunc ornare mi at augue vulputate, eu venenatis magna mollis. Nunc sed posuere dui, et varius nulla. Sed mollis nibh augue, eget scelerisque eros ornare nec.

1. Donec pretium et quam a cursus. Ut sollicitudin tempus urna et mollis.
2. Aliquam et aliquam turpis, sed fermentum mauris. Nulla eget ex diam.
3. Donec eget tellus pharetra, semper neque eget, rutrum diam 1. lépés.

Praesent porta, metus eget eleifend consequat, eros ligula eleifend ex, a pellentesque mi est vitae urna. Vivamus turpis nunc, iaculis non leo eget, mattis vulputate tellus. Maecenas rutrum eros sem, pharetra interdum nulla porttitor sit amet. In vitae viverra ante. Maecenas sit amet placerat orci, sed tincidunt velit. Vivamus mattis, enim vel suscipit elementum, quam odio venenatis elit[1], et mollis nulla nunc a risus. Praesent purus magna, tristique sed lacus sit amet, convallis malesuada magna.

Vestibulum venenatis malesuada enim, ac auctor erat vestibulum et. Phasellus id purus a leo suscipit accumsan.

Orci varius natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Nullam interdum rhoncus nisl, vel pharetra arcu euismod sagittis. Vestibulum ac turpis auctor, viverra turpis at, tempus tellus.

Morbi dignissim erat ut rutrum aliquet. Nulla eu rutrum urna. Integer non urna at mauris scelerisque rutrum sed non turpis.

2.1.1. Szoros térközű felsorolások

Phasellus ultricies, sapien sit amet ultricies placerat, velit purus viverra ligula, id consequat ipsum odio imperdiet enim:

1. Maecenas eget lobortis leo.
2. Donec eget libero enim.
3. In eu eros a eros lacinia maximus ullamcorper eget augue.

In quis turpis metus. Proin maximus nibh et massa eleifend, a feugiat augue porta. Sed eget est purus. Duis in placerat leo. Donec pharetra eros nec enim convallis:

[1] Phasellus faucibus varius purus, nec tristique enim porta vitae.

- Pellentesque odio lacus.
- Maximus ut nisl auctor.
- Sagittis vulputate lorem.

Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Sed lorem libero, dignissim vitae gravida a, ornare vitae est.

Cras maximus massa commodo pellentesque viverra.

Morbi sit amet ante risus. Aliquam nec sollicitudin mauris

Ut aliquam rhoncus sapien luctus viverra arcu iaculis posuere

2.2. Képek, ábrák

Aliquam vehicula luctus mi a pretium. Nulla quam neque, maximus nec velit in, aliquam mollis tortor. Aliquam erat volutpat. Curabitur vitae laoreet turpis. Integer id diam ligula. Nulla sodales purus id mi consequat, eu venenatis odio pharetra. Cras a arcu quam. Suspendisse augue risus, pulvinar a turpis et, commodo aliquet turpis. Nulla aliquam scelerisque mi eget pharetra. Mauris sed posuere elit, ac lobortis metus. Proin lacinia sit amet diam sed auctor. Nam viverra orci id sapien sollicitudin, a aliquam lacus suscipit, 2.1. ábra:

2.1. ábra. Quisque ac tincidunt leo

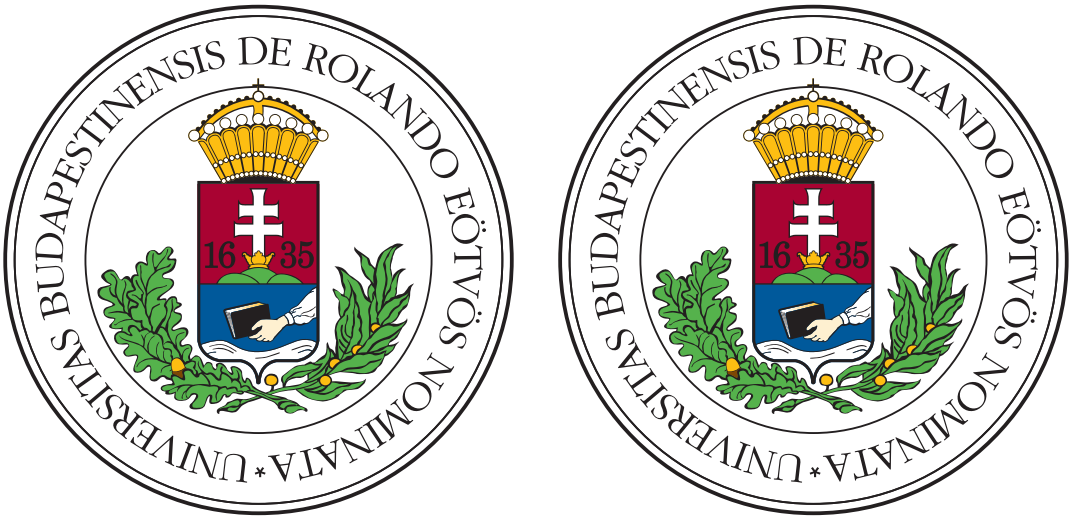
2.2.1. Képek szegélyezése

Ut aliquet nec neque eget fermentum. Cras volutpat tellus sed placerat elementum. Quisque neque dui, consectetur nec finibus eget, blandit id purus. Nam eget ipsum non nunc placerat interdum.

2.2. ábra. Quisque ac tincidunt leo

2.2.2. Képek csoportosítása

In non ipsum fermentum urna feugiat rutrum a at odio. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Nulla tincidunt mattis nisl id suscipit. Sed bibendum ac felis sed volutpat. Nam pharetra nisi nec facilisis faucibus. Aenean tristique nec libero non commodo. Nulla egestas laoreet tempus. Nunc eu aliquet nulla, quis vehicula dui. Proin ac risus sodales, gravida nisi vitae, efficitur neque, 2.3. ábra:



(a) Vestibulum quis mattis urna

(b) Donec hendrerit quis dui sit amet venenatis

2.3. ábra. Aenean porttitor mi volutpat massa gravida

Nam et nunc eget elit tincidunt sollicitudin. Quisque ligula ipsum, tempor vitae tortor ut, commodo rhoncus diam. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Phasellus vehicula quam dui, eu convallis metus porta ac.

2.3. Táblázatok

Nam magna ex, euismod nec interdum sed, sagittis nec leo. Nam blandit massa bibendum mattis tristique. Phasellus tortor ligula, sodales a consectetur vitae, placerat vitae dolor. Aenean consequat in quam ac mollis.

Phasellus tortor	**Aenean consequat**
Sed malesuada	Aliquam aliquam velit in convallis ultrices.
Purus sagittis	Quisque lobortis eros vitae urna lacinia euismod.
Pellentesque	Curabitur ac lacus pellentesque, eleifend sem ut, placerat enim. Ut auctor tempor odio ut dapibus.

2.1. táblázat. Maecenas tincidunt non justo quis accumsan

2.3.1. Sorok és oszlopok egyesítése

Mauris a dapibus lectus. Vestibulum commodo nibh ante, ut maximus magna eleifend vel. Integer vehicula elit non lacus lacinia, vitae porttitor dolor ultrices. Vivamus gravida faucibus efficitur. Ut non erat quis arcu vehicula lacinia. Nulla felis mauris, laoreet sed malesuada in, euismod et lacus. Aenean at finibus ipsum. Pellentesque dignissim elit sit amet lacus congue vulputate.

Quisque	**Suspendisse**		**Aliquam**		**Vivamus**	
	Proin	Nunc	Proin	Nunc	Proin	Nunc
Leo	2,80 MB	100%	232 KB	8,09%	248 KB	8,64%
Vel	9,60 MB	100%	564 KB	5,74%	292 KB	2,97%
Auge	78,2 MB	100%	52,3 MB	66,88%	3,22 MB	4,12%

2.2. táblázat. Vivamus ac arcu fringilla, fermentum neque sed, interdum erat. Mauris bibendum mauris vitae enim mollis, et eleifend turpis aliquet.

2.3.2. Több oldalra átnyúló táblázatok

Nunc porta placerat leo, sit amet porttitor dui porta molestie. Aliquam at fermentum mi. Maecenas vitae lorem at leo tincidunt volutpat at nec tortor. Vivamus semper lacus eu diam laoreet congue. Vivamus in ipsum risus. Nulla ullamcorper finibus mauris non aliquet. Vivamus elementum rhoncus ex ut porttitor.

Praesent aliquam mauris enim	
Suspendisse potenti	*Lorem ipsum dolor sit amet*
Praesent	Nulla ultrices et libero sit amet fringilla. Nunc scelerisque ante tempus sapien placerat convallis.
Luctus	Integer hendrerit erat massa, non hendrerit risus convallis at. Curabitur ultrices, justo in imperdiet condimentum, neque tortor luctus enim, luctus posuere massa erat vitae nibh.
Egestas	Duis fermentum feugiat augue in blandit. Mauris a tempor felis. Pellentesque ultricies tristique dignissim. Pellentesque aliquam semper tristique. Nam nec egestas dolor. Vestibulum id elit quis enim fringilla tempor eu a mauris. Aliquam vitae lacus tellus. Phasellus mauris lectus, aliquam id leo eget, auctor dapibus magna. Fusce lacinia felis ac elit luctus luctus.
Dignissim	Praesent aliquam mauris enim, vestibulum posuere massa facilisis in. Suspendisse potenti. Nam quam purus, rutrum eu augue ut, varius vehicula tellus. Fusce dui diam, aliquet sit amet eros at, sollicitudin facilisis quam. Phasellus tempor metus vel augue gravida pretium. Proin aliquam aliquam blandit. Nulla id tempus mi. Fusce in aliquam tortor.
Pellentesque	Donec felis nibh, imperdiet a arcu non, vehicula gravida nibh. Quisque interdum sapien eu massa commodo, ac elementum felis faucibus.
Molestie	Cras ullamcorper tellus et auctor ultricies. Maecenas tincidunt euismod lectus nec venenatis. Suspendisse potenti. Pellentesque pretium nunc ut euismod cursus. Nam venenatis condimentum quam. Curabitur suscipit efficitur aliquet. Interdum et malesuada fames ac ante ipsum primis in faucibus.

Suspendisse potenti	*Lorem ipsum dolor sit amet*
Vivamus semper	In purus purus, faucibus eu libero vulputate, tristique sodales nunc. Nulla ut gravida dolor. Fusce vel pellentesque mi, vel efficitur eros. Nunc vitae elit tellus. Sed vestibulum auctor consequat.
Condimentum	Nulla scelerisque, leo et facilisis pretium, risus enim cursus turpis, eu suscipit ipsum ipsum in mauris. Praesent eget pulvinar ipsum, suscipit interdum nunc. Nam varius massa ut justo ullamcorper sollicitudin. Vivamus facilisis suscipit neque, eu fermentum risus. Ut at mi mauris.

2.3. táblázat. Praesent ullamcorper consequat tellus ut eleifend

3. fejezet

Fejlesztői dokumentáció

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis nibh leo, dapibus in elementum nec, aliquet id sem. Suspendisse potenti. Nullam sit amet consectetur nibh. Donec scelerisque varius turpis at tincidunt.

3.1. Forráskódok

Nulla sodales purus id mi consequat, eu venenatis odio pharetra. Cras a arcu quam. Suspendisse augue risus, pulvinar a turpis et, commodo aliquet turpis. Nulla aliquam scelerisque mi eget pharetra. Mauris sed posuere elit, ac lobortis metus. Proin lacinia sit amet diam sed auctor. Nam viverra orci id sapien sollicitudin, a aliquam lacus suscipit. Quisque ac tincidunt leo 3.1. és 3.2. forráskód:

```
#include <stdio>

int main()
{
  int c;
  std::cout << "Hello World!" << std::endl;

  std::cout << "Press any key to exit." << std::endl;
  std::cin >> c;

  return 0;
}
```

3.1. forráskód. Hello World in C++

```
using System;
namespace HelloWorld
{
  class Hello
  {
    static void Main()
    {
      Console.WriteLine("Hello World!");

      Console.WriteLine("Press any key to exit.");
      Console.ReadKey();
    }
  }
}
```

3.2. forráskód. Hello World in C#

3.1.1. Algoritmusok

Az 1. algoritmus egy általános elágazás és korlátozás algoritmust (*Branch and Bound algorithm*) mutat be. A 3. lépésben egy megfelelő kiválasztási szabályt kell alkalmazni. Példa forrása: Acta Cybernetica (ez egy hiperlink).

1. algoritmus A general interval B&B algorithm

Funct IBB(S, f)

Set the working list $\mathcal{L}_W := \{S\}$ and the final list $\mathcal{L}_Q := \{\}$
while ($\mathcal{L}_W \neq \emptyset$) **do**
    Select an interval X from $\mathcal{L}_W$ ▷ Selection rule
    Compute $lbf(X)$ ▷ Bounding rule
    if X cannot be eliminated **then** ▷ Elimination rule
        Divide X into X^j, $j = 1, \ldots, p$, subintervals ▷ Division rule
        for $j = 1, \ldots, p$ **do**
            if X^j satisfies the termination criterion **then** ▷ Termination rule
                Store X^j in $\mathcal{L}_W$
            else
                Store X^j in $\mathcal{L}_W$
            end if
        end for
    end if
end while
return $\mathcal{L}_Q$

3.2. Classfájltól, benne levő metódus futtatásáig

A fő osztály a `ClassFile`, ez felel számos dologért, többek között egy Classfájl beolvasáért, a megfelelő adattagok beállításával. A `ClassFile` osztálynak egy konstruktora van, mégpedig:

```
1 public ClassFile(String fileName, String[] mainArgs)
```

Tehát az első paraméter a beolvasandó classfájl neve, a második pedig a `main` metódusnak adott argumentumok.

Az implementáció alapján nem kötött a `main` metódus használata belépési pontként, tehát a 2. argumentum lehet `null` is.

A konstruktor meghívása egyidejüleg meghívja a `readClassFile` függvényt is:

```
1 public void readClassFile(String fileName)
```

Ez a függvény egy adott fájlnévre beolvassa a Classfájlban tárolt adatokat megfelelő változókba. (Ezen felül egy `VALID_CLASS_FILE` változót is beállít; feltételezük hogy ha a mágikus szám `(CA FE BA BE)` megtalálható a fájl elején, akkor az adott fájl egy valid Classfájl, ellenkező esetben egy `InvalidClassFileException`-t dob a beolvasó függvény.)

A beolvasás után (tehát az objektum létrehozása után) érdemes a belépési függvényt (általában `main`) megkeresni a `findMethodsByName` metódussal:

```
1 public Method_Info findMethodsByName(String methodName)
```

Ez egy adott függvénynévre a megfelelő nevü metódust visszaadja a beolvasott fájlból (ha nem talál ilyet akkor `null`-t ad vissza). Egy példa a használatára:

```
1 ClassFile CLASS_FILE = new ClassFile("Main.class", null);
2 Method_Info method = CLASS_FILE.findMethodsByName("main");
```

A függvény megtalálása után ajánlott a `Code` attribútumot megtalálni, ebben, többek között, található a futtatandó bytecode is. A segédfüggvény erre a `findAttributesByName`:

```
public List<Attribute_Info>
    findAttributesByName(List<Attribute_Info> attributes, String
    attributeName)
```

Mivel egy attribútumból több is lehet, egy listát kapunk vissza (a `Code`-ból csak egy lesz), bemeneti paraméterként az attribútumnév mellett a megfelelü függvény attribútumait is át kell adnuk, például:

```
List<Attribute_Info> attributes =
    CLASS_FILE.findAttributesByName(method.attributes, "Code");
```

(Ha nem talál ilyen nevezetű attribútumot akkor üres listát ad vissza.)

A megfelelüen beolvasott attribútum után, a megtalált attribútumok között ajanlott végigmenni, a `List` implementálja az `Iterable`-t, így egy for ciklussal elegánsan megtehetjük ezt:

```
for (Attribute_Info attribute : attributes)
```

Mivel `Code` attribútumokról beszélünk, ezért a következő ajánlott dolog hogy ebből az attribútumból olvassuk be az adatokat. Ehhez a `Code_Attribute_Helper` osztály `readCodeAttributes` metódusa megfelelő:

```
public static Code_Attribute readCodeAttributes(Attribute_Info
    attribute) throws IOException
```

A függvény egy attribútumot vár (például az előbbi kódrészlet `attribute` változóját), majd pedig beolvassa a specifikációnak megfelelően a `Code_Attribute`-ot, és visszaadja azt, ha valamiért nem sikerült a beolvasás akkor `IOException`-t dob a függvény.

```
Code_Attribute codeAttribute =
    Code_Attribute_Helper.readCodeAttributes(attribute);
```

Ezt a beolvasott attribútumot a `ClassFile` osztály fel tudja használni az `executeCode` metódusával, mely egy `byte[]` változót vár bemeneti paraméterként, ami a `Code_Attribute` része:

```
public Pair<Class<?>, Object> executeCode(byte[] code)
  throws IOException, ClassNotFoundException, NoSuchFieldException,
   ↪  IllegalAccessException,
  NoSuchMethodException, SecurityException, InstantiationException,
   ↪  IllegalArgumentException,
  InvocationTargetException, Throwable
```

A reflekció miatt számos hibát dob vissza a függvény, ha nem helyes a kód formátuma akkor `IOException`-t dob a függvény, a `Throwable` az `ATHROW` bytecode instrukció miatt szükséges (ekkor egy hibát dob vissza a metódusunk). Visszatérési értéke `Pair<Class<?>, Object>`, a számos `RETURN` utasítás miatt (ezeket a stack-en szükséges elhelyezünk) Példa a használatára:

```
CLASS_FILE.executeCode(codeAttribute.code);
```

Ezzel el is jutottunk egy Classfájl beolvasásától, az abban lévő adott függvény bytecodejának futtatásáig, több teendőnk nincsen, a program az adott függvényben levő külön függvényhívásokat automatikusan elvégzi.

A teljes példakód:

```
ClassFile CLASS_FILE = new ClassFile("Main.class", null);
Method_Info method = CLASS_FILE.findMethodsByName("main");
List<Attribute_Info> attributes =
 ↪  CLASS_FILE.findAttributesByName(method.attributes, "Code");

for (Attribute_Info attribute : attributes) {
  Code_Attribute codeAttribute =
   ↪  Code_Attribute_Helper.readCodeAttributes(attribute);
  CLASS_FILE.executeCode(codeAttribute.code);
}
```

3.3. Pár minta Classfájl felépítése

A legegyszerűbb Classfájl ami értelmes, viszont nem futattható:

```
CA FE BA BE 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00
```

Java kódban ennek megfelelője az üres fájl:

```
1
```

Classfájl formátumának magyarázata:

- CA FE BA BE: Mágikus szám, amely minden Class fájl elején megtalálható
- 00 00 00 00: Class fájl `Minor` és `Major` verziószáma, egy táblázatnak megfelelően a fordítóprogram verziója
- 00 00: A Constant Pool mérete (+1, mivel 1-től indexelt, itt nem számít)
- 00 00: Hozzáférési zászlók ()
- 00 00: `This` osztály indexe a Constant Pool-ban
- 00 00: `Super` osztály indexe a Constant Pool-ban
- 00 00: Interfészek száma
- 00 00: Adattagok száma
- 00 00: Függvények száma
- 00 00: Osztály attribútumainak száma

A legegyszerűbb Classfájl amit a *Jabyinja* program le tud futtatni (a beépített `java` program nem képes ezt lefuttatni, mivel nincsenek benne osztályok, a JVM specifikáció alapján az osztályok elhanyagolhatóak):

CA	FE	BA	BE	00	00	00	00	00	04	01	00	04	43	6F
64	65	01	00	04	6D	61	69	6E	01	00	03	28	29	56
00	21	00	00	00	00	00	00	00	00	00	01	00	09	00
02	00	03	00	01	00	01	00	00	00	0D	00	00	00	00
00	00	00	01	B1	00	00	00	00	00	00				

Java kód megfelelője:

```
1 public static void main() {
2     return;
3 }
```

Classfájl formátumának magyarázata:

- CA FE BA BE: Mágikus szám, amely minden Classfájl elején megtalálható
- 00 00 00 00: Classfájl `Minor` és `Major` verziószáma, egy táblázatnak megfelelően a `javac` fordítóprogram verziója
- 00 04: A Constant Pool mérete (+1, mivel 1-től indexelt)
- 01 00 04 43 6F 64 65 01 00 04 6D 61 69 6E 01 00 03 28 29 56: Constant Pool
  - 01 00 04 43 6F 64 65

    01: Constant Pool Info érték `(CONSTANT_Utf8)`

    00 04: 4 hosszú

    43 6F 64 65: A `CONSTANT_Utf8` értéke: `Code`
  - 01 00 04 6D 61 69 6E

    01: Constant Pool Info érték `(CONSTANT_Utf8)`

    00 04: 4 hosszú

    6D 61 69 6E: A `CONSTANT_Utf8` értéke: `main`
  - 01 00 03 28 29 56

    01: Constant Pool Info érték `(CONSTANT_Utf8)`

    00 03: 3 hosszú

    28 29 56: A `CONSTANT_Utf8` értéke: `()V`
- 00 21: Hozzáférési zászlók `(Public, Super)` - elhanyagolhatóak ebben az esetben
- 00 00: `This` osztály indexe a Constant Pool-ban
- 00 00: `Super` osztály indexe a Constant Pool-ban
- 00 00: Interfészek száma

- 00 00: Adattagok száma
- 00 01: Függvények száma
- 00 09 00 02 00 03 00 01 00 01 00 00 00 0D 00 00 00 00 00 00 00 01 B1 00 00 00 00: Függvények
  - 00 09 00 02 00 03 00 01 00 01 00 00 00 0D 00 00 00 00 00 00 00 01 B1 00 00 00 00

    00 09: Hozzáférési zászlók `(Public, Static)`

    00 02: Constant Poolban lévő indexe a függvénynek: `main`

    00 03: Függvény leírása (bemeneti paraméterek, visszatérési érték): `()V`

    00 01: Függvény attribútumainak száma

    00 01 00 00 00 0D 00 00 00 00 00 00 00 01 B1 00 00 00 00: Attribútumok
    - 00 01 00 00 00 0D 00 00 00 00 00 00 00 01 B1 00 00 00 00

      00 01: Constant Pool-ban lévő indexe az attribútumnak: `Code`

      00 00 00 0D: Attribútum hossza (`0D` = 13 bájt)

      00 00 00 00 00 00 00 01 B1 00 00 00 00: Attribútum
      - 00 00 00 00 00 00 00 01 B1 00 00 00 00

        00 00: Stack mérete

        00 00: Lokális változók száma

        00 00 00 01: Kód hossza

        B1: Kód `(B1 = return)`

        00 00: Kivételek száma

        00 00: Attribútum attribútumainak száma
- 00 00: Osztály attribútumainak száma

3.4. JVM Bytecode utasítások

Az összes bytecode utasítás implementálva van, ezekről az alábbi táblázatban röviden pár dolog le van írva, nemlegesen a hex kódjuk, extra értékek utánuk, illetve hogy a stack-et hogyan változtatják

Bytecode utasítások			
Neve	*HEX*	*Paraméterek*	*Stack*
NOP	00		
ACONST_NULL	01		→ NULL

Neve	*HEX*	*Paraméterek*	*Stack*
ICONST_M1	02		→ -1
ICONST_0	03		→ 0
ICONST_1	04		→ 1
ICONST_2	05		→ 2
ICONST_3	06		→ 3
ICONST_4	07		→ 4
ICONST_5	08		→ 5
LCONST_0	09		→ 0L
LCONST_1	0A		→ 1L
FCONST_0	0B		→ 0.0f
FCONST_1	0C		→ 1.0f
FCONST_2	0D		→ 2.0f
DCONST_0	0E		→ 0.0
DCONST_1	0F		→ 1.0
BIPUSH	10	`u8 value`	→ `value`
SIPUSH	11	`u16 value`	→ `value`
LDC	12	`u8 index`	→ `CONSTANT_POOL[index]`
LDC_W	13	`u16 index`	→ `CONSTANT_POOL[index]`
LDC2_W	14	`u16 index`	→ `CONSTANT_POOL[index]`
ILOAD	15	`u8 index`	→ `LOCAL[index]`
LLOAD	16	`u8 index`	→ `LOCAL[index]`
FLOAD	17	`u8 index`	→ `LOCAL[index]`
DLOAD	18	`u8 index`	→ `LOCAL[index]`
ALOAD	19	`u8 index`	→ `LOCAL[index]`
ILOAD_0	1A		→ `LOCAL[0]`
ILOAD_1	1B		→ `LOCAL[1]`
ILOAD_2	1C		→ `LOCAL[2]`
ILOAD_3	1D		→ `LOCAL[3]`
LLOAD_0	1E		→ `LOCAL[0]`
LLOAD_1	1F		→ `LOCAL[1]`
LLOAD_2	20		→ `LOCAL[2]`
LLOAD_3	21		→ `LOCAL[3]`

Neve	*HEX*	*Paraméterek*	*Stack*
FLOAD_0	22		→ `LOCAL[0]`
FLOAD_1	23		→ `LOCAL[1]`
FLOAD_2	24		→ `LOCAL[2]`
FLOAD_3	25		→ `LOCAL[3]`
DLOAD_0	26		→ `LOCAL[0]`
DLOAD_1	27		→ `LOCAL[1]`
DLOAD_2	28		→ `LOCAL[2]`
DLOAD_3	29		→ `LOCAL[3]`
ALOAD_0	2A		→ `LOCAL[0]`
ALOAD_1	2B		→ `LOCAL[1]`
ALOAD_2	2C		→ `LOCAL[2]`
ALOAD_3	2D		→ `LOCAL[3]`
IALOAD	2E		`arrayref, index → value`
LALOAD	2F		`arrayref, index → value`
FALOAD	30		`arrayref, index → value`
DALOAD	31		`arrayref, index → value`
AALOAD	32		`arrayref, index → value`
BALOAD	33		`arrayref, index → value`
CALOAD	34		`arrayref, index → value`
SALOAD	35		`arrayref, index → value`
ISTORE	36	`u8 index`	`value →`
LSTORE	37	`u8 index`	`value →`
FSTORE	38	`u8 index`	`value →`
DSTORE	39	`u8 index`	`value →`
ASTORE	3A	`u8 index`	`objectref →`
ISTORE_0	3B		`value →`
ISTORE_1	3C		`value →`
ISTORE_2	3D		`value →`
ISTORE_3	3E		`value →`
LSTORE_0	3F		`value →`
LSTORE_1	40		`value →`
LSTORE_2	41		`value →`

Neve	*HEX*	*Paraméterek*	*Stack*
LSTORE_3	42		`value` →
FSTORE_0	43		`value` →
FSTORE_1	44		`value` →
FSTORE_2	45		`value` →
FSTORE_3	46		`value` →
DSTORE_0	47		`value` →
DSTORE_1	48		`value` →
DSTORE_2	49		`value` →
DSTORE_3	4A		`value` →
ASTORE_0	4B		`value` →
ASTORE_1	4C		`value` →
ASTORE_2	4D		`value` →
ASTORE_3	4E		`value` →
IASTORE	4F		`arrayref, index, value` →
LASTORE	50		`arrayref, index, value` →
FASTORE	51		`arrayref, index, value` →
DASTORE	52		`arrayref, index, value` →
AASTORE	53		`arrayref, index, value` →
BASTORE	54		`arrayref, index, value` →
CASTORE	55		`arrayref, index, value` →
SASTORE	56		`arrayref, index, value` →
POP	57		`value` →
POP2	58		`{value2, value1}` →
DUP	59		`value` → `value, value`
DUP_X1	5A		`{value2, value1}` → `value1, {value2, value1}`
DUP_X2	5B		`value3, value2, value1` → `value1, value3, value2, value1`
DUP2	5C		`{value2, value1}` → `{value2, value1}, {value2, value1}`
DUP2_X1	5D		`value3, {value2, value1}` →

Neve	*HEX*	*Paraméterek*	*Stack*
			`{value2, value1}, value3, {value2, value1}`
DUP2_X2	5E		`{value4, value3}, {value2, value1}` → `{value2, value1}, {value4, value3}, {value2, value1}`
SWAP	5F		`value2, value1` → `value1, value2`
IADD	60		`value1, value2` → `result`
LADD	61		`value1, value2` → `result`
FADD	62		`value1, value2` → `result`
DADD	63		`value1, value2` → `result`
ISUB	64		`value1, value2` → `result`
LSUB	65		`value1, value2` → `result`
FSUB	66		`value1, value2` → `result`
DSUB	67		`value1, value2` → `result`
IMUL	68		`value1, value2` → `result`
LMUL	69		`value1, value2` → `result`
FMUL	6A		`value1, value2` → `result`
DMUL	6B		`value1, value2` → `result`
IDIV	6C		`value1, value2` → `result`
LDIV	6D		`value1, value2` → `result`
FDIV	6E		`value1, value2` → `result`
DDIV	6F		`value1, value2` → `result`
IREM	70		`value1, value2` → `result`
LREM	71		`value1, value2` → `result`
FREM	72		`value1, value2` → `result`
DREM	73		`value1, value2` → `result`
INEG	74		`value` → `result`
LNEG	75		`value` → `result`
FNEG	76		`value` → `result`
DNEG	77		`value` → `result`
ISHL	78		`value1, value2` → `result`

Neve	*HEX*	*Paraméterek*	*Stack*
LSHL	79		`value1, value2 → result`
ISHR	7A		`value1, value2 → result`
LSHR	7B		`value1, value2 → result`
IUSHR	7C		`value1, value2 → result`
LUSHR	7D		`value1, value2 → result`
IAND	7E		`value1, value2 → result`
LAND	7F		`value1, value2 → result`
IOR	80		`value1, value2 → result`
LOR	81		`value1, value2 → result`
IXOR	82		`value1, value2 → result`
LXOR	83		`value1, value2 → result`
IINC	84	`u8 index,` `u8 const`	`.. → ..`
I2L	85		`value → result`
I2F	86		`value → result`
I2D	87		`value → result`
L2I	88		`value → result`
L2F	89		`value → result`
L2D	8A		`value → result`
F2I	8B		`value → result`
F2L	8C		`value → result`
F2D	8D		`value → result`
D2I	8E		`value → result`
D2L	8F		`value → result`
D2F	90		`value → result`
I2B	91		`value → result`
I2C	92		`value → result`
I2S	93		`value → result`
LCMP	94		`value1, value2 → result`
FCMPL	95		`value1, value2 → result`
FCMPG	96		`value1, value2 → result`
DCMPL	97		`value1, value2 → result`

Neve	*HEX*	*Paraméterek*	*Stack*
DCMPG	98		`value1, value2 → result`
IFEQ	99	`u16 branch`	`value →`
IFNE	9A	`u16 branch`	`value →`
IFLT	9B	`u16 branch`	`value →`
IFGE	9C	`u16 branch`	`value →`
IFGT	9D	`u16 branch`	`value →`
IFLE	9E	`u16 branch`	`value →`
IF_ICMPEQ	9F	`u16 branch`	`value1, value2 →`
IF_ICMPNE	A0	`u16 branch`	`value1, value2 →`
IF_ICMPLT	A1	`u16 branch`	`value1, value2 →`
IF_ICMPGE	A2	`u16 branch`	`value1, value2 →`
IF_ICMPGT	A3	`u16 branch`	`value1, value2 →`
IF_ICMPLE	A4	`u16 branch`	`value1, value2 →`
IF_ACMPEQ	A5	`u16 branch`	`value1, value2 →`
IF_ACMPNE	A6	`u16 branch`	`value1, value2 →`
GOTO	A7	`u16 branch`	`.. → ..`
JSR	A8	`u16 branch`	`→ address`
RET	A9	`u8 index`	`.. → ..`
TABLESWITCH	AA	`[0-3] byte padding,` `u32 default,` `u32 low,` `u32 high,` `jump offsets`	`index →`
LOOKUPSWITCH	AB	`[0-3] byte padding,` `u32 default,` `u32 npairs,` `match-offset pairs`	`key →`
IRETURN	AC		`value →`
LRETURN	AD		`value →`

Neve	*HEX*	*Paraméterek*	*Stack*
FRETURN	AE		`value` →
DRETURN	AF		`value` →
ARETURN	B0		`objectref` →
RETURN	B1		`..` →
GETSTATIC	B2	`u16 index`	→ `value`
PUTSTATIC	B3	`u16 index`	`value` →
GETFIELD	B4	`u16 index`	`objectref` → `value`
PUTFLIED	B5	`u16 index`	`objectref, value` →
INVOKEVIRTUAL	B6	`u16 index`	`objectref, [args]` → `result`
INVOKESPECIAL	B7	`u16 index`	`objectref, [args]` → `result`
INVOKESTATIC	B8	`u16 index`	`[args]` → `result`
INVOKEINTERFACE	B9	`u16 index` `u8 count, 0`	`objectref, [args]` → `result`
INVOKEDYNAMIC	BA	`u16 index,` `0, 0`	`[args]` → `result`
NEW	BB	`u16 index`	→ `objectref`
NEWARRAY	BC	`u8 atype`	`count` → `arrayref`
ANEWARRAY	BD	`u16 index`	`count` → `arrayref`
ARRAYLENGTH	BE	`u16 index`	`arrayref` → `length`
ATHROW	BF		`objectref` → `objectref`
CHECKCAST	C0	`u16 index`	`objectref` → `objectref`
INSTANCEOF	C1	`u16 index`	`objectref` → `result`
MONITORENTER	C2	`u16 index`	`objectref` →
MONITOREXIT	C3	`u16 index`	`objectref` →

WIDE	C4	`u8 opcode`
		`u16 index`
		`vagy`
		`u8 opcode,`
		`u16 index,`
		`u16 count`
			`mint az adott utasitas`

MULTIANEWARRAY	C5	`u16 index,` `u8 dimensions`	`[counts]` → `arrayref`

Neve	*HEX*	*Paraméterek*	*Stack*
IFNULL	C6	`u16 branch`	`value` →
IFNONNULL	C7	`u16 branch`	`value` →
GOTO_W	C8	`u32 branch`	`..` → `..`
JSR_W	C9	`u32 branch`	→ `address`

3.1. táblázat. A JVM Bytecode különböző utasításai

4. fejezet

Összegzés

Lorem ipsum dolor sit amet, consectetur adipiscing elit. In eu egestas mauris. Quisque nisl elit, varius in erat eu, dictum commodo lorem. Sed commodo libero et sem laoreet consectetur. Fusce ligula arcu, vestibulum et sodales vel, venenatis at velit. Aliquam erat volutpat. Proin condimentum accumsan velit id hendrerit. Cras egestas arcu quis felis placerat, ut sodales velit malesuada. Maecenas et turpis eu turpis placerat euismod. Maecenas a urna viverra, scelerisque nibh ut, malesuada ex.

Aliquam suscipit dignissim tempor. Praesent tortor libero, feugiat et tellus porttitor, malesuada eleifend felis. Orci varius natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Nullam eleifend imperdiet lorem, sit amet imperdiet metus pellentesque vitae. Donec nec ligula urna. Aliquam bibendum tempor diam, sed lacinia eros dapibus id. Donec sed vehicula turpis. Aliquam hendrerit sed nulla vitae convallis. Etiam libero quam, pharetra ac est nec, sodales placerat augue. Praesent eu consequat purus.

Köszönyetnyilvánítás

Petes Márton (ELTE IK PTI BSc): Az elekadásaim során elképesztően sok segítséget nyújtott, nélküle nem tudom hogy meglett volna-e a szakdolgozat

A. függelék

Szimulációs eredmények

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque facilisis in nibh auctor molestie. Donec porta tortor mauris. Cras in lacus in purus ultricies blandit. Proin dolor erat, pulvinar posuere orci ac, eleifend ultrices libero. Donec elementum et elit a ullamcorper. Nunc tincidunt, lorem et consectetur tincidunt, ante sapien scelerisque neque, eu bibendum felis augue non est. Maecenas nibh arcu, ultrices et libero id, egestas tempus mauris. Etiam iaculis dui nec augue venenatis, fermentum posuere justo congue. Nullam sit amet porttitor sem, at porttitor augue. Proin bibendum justo at ornare efficitur. Donec tempor turpis ligula, vitae viverra felis finibus eu. Curabitur sed libero ac urna condimentum gravida. Donec tincidunt neque sit amet neque luctus auctor vel eget tortor. Integer dignissim, urna ut lobortis volutpat, justo nunc convallis diam, sit amet vulputate erat eros eu velit. Mauris porttitor dictum ante, commodo facilisis ex suscipit sed.

Sed egestas dapibus nisl, vitae fringilla justo. Donec eget condimentum lectus, molestie mattis nunc. Nulla ac faucibus dui. Nullam a congue erat. Ut accumsan sed sapien quis porttitor. Ut pellentesque, est ac posuere pulvinar, tortor mauris fermentum nulla, sit amet fringilla sapien sapien quis velit. Integer accumsan placerat lorem, eu aliquam urna consectetur eget. In ligula orci, dignissim sed consequat ac, porta at metus. Phasellus ipsum tellus, molestie ut lacus tempus, rutrum convallis elit. Suspendisse arcu orci, luctus vitae ultricies quis, bibendum sed elit. Vivamus at sem maximus leo placerat gravida semper vel mi. Etiam hendrerit sed massa ut lacinia. Morbi varius libero odio, sit amet auctor nunc interdum sit amet.

Aenean non mauris accumsan, rutrum nisi non, porttitor enim. Maecenas vel tortor ex. Proin vulputate tellus luctus egestas fermentum. In nec lobortis risus,

sit amet tincidunt purus. Nam id turpis venenatis, vehicula nisl sed, ultricies nibh. Suspendisse in libero nec nisi tempor vestibulum. Integer eu dui congue enim venenatis lobortis. Donec sed elementum nunc. Nulla facilisi. Maecenas cursus id lorem et finibus. Sed fermentum molestie erat, nec tempor lorem facilisis cursus. In vel nulla id orci fringilla facilisis. Cras non bibendum odio, ac vestibulum ex. Donec turpis urna, tincidunt ut mi eu, finibus facilisis lorem. Praesent posuere nisl nec dui accumsan, sed interdum odio malesuada.

Ábrák jegyzéke

Táblázatok jegyzéke

Algoritmusjegyzék

Forráskódjegyzék