Metode Avansate de Programare
Limbajul Java. Dezvoltarea software (Introducere)

Arthur Molnar
arthur.molnar@ubbcluj.ro

Universitatea Babeș-Bolyai

2023

Privire de ansamblu

Introducere în Java I

Avantaje

- Ușor de învățat și utilizat (n.a. mai puțin decât Python, dar mai mult decât C++)
- Orientat Obiect și Funcțional (începând cu Java 8)
- Independent de platformă
- API-uri bogate
- Robust (compilat, utilizare obligatorie a excepțiilor, garbage collection)
- Sigur, distribuit, multi-thread

Dezavantaje: mai lent și consumator de memorie comparat cu limbaje de nivel mai jos (C sau C++)

Java vs. C++

Java	**C++**
Suportă interfețe	Nu are noțiunea explicită a unei *interfețe*
Nu suportă moștenirea multiplă	Suportă moștenirea multiplă
Polimorfism automat	Polimorfism explicit
Sistemul are mai multe responsabilități	Programatorul are mai multe responsabilități
Nu sunt pointeri	Pointeri!
Codul e scris în cadrul unei clase	Putem avea funcții și date în afara unei clase
Independent de platformă	Executabilele compilate depind de platformă

Compilarea și executarea unui program

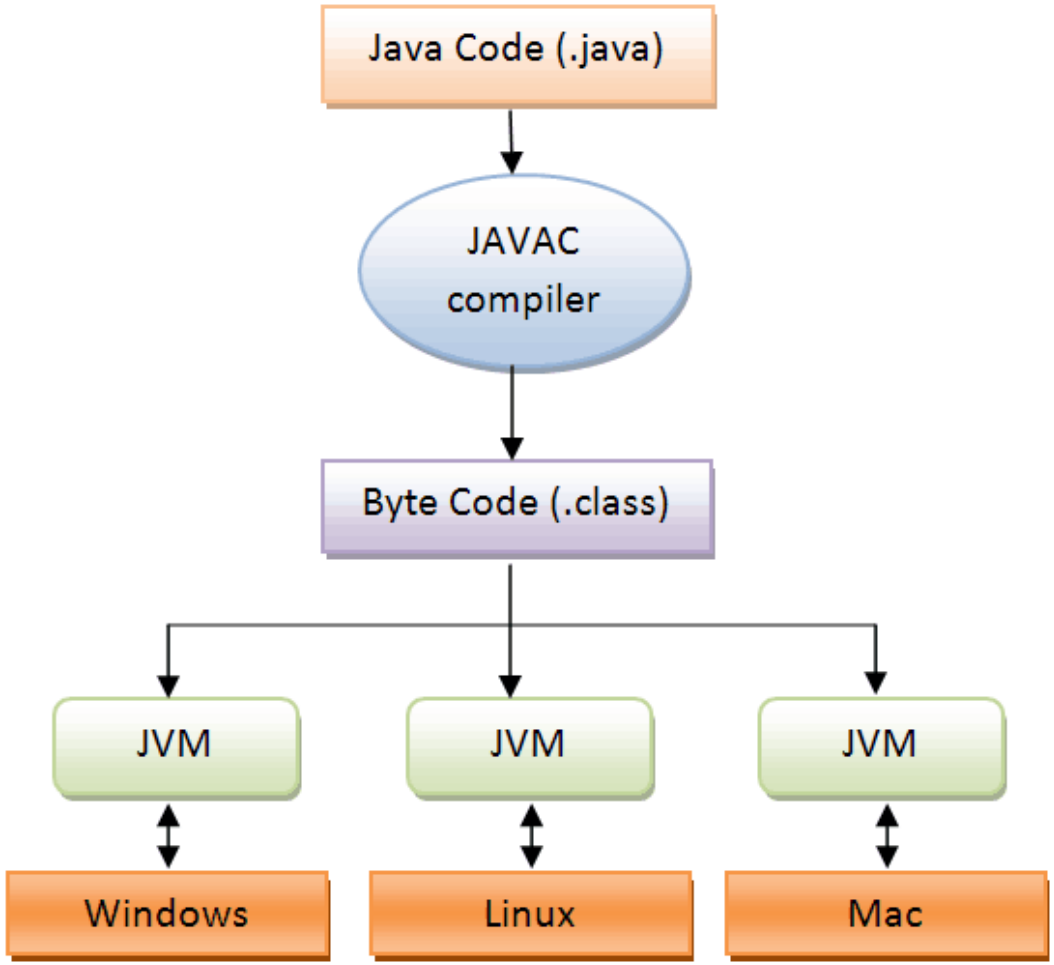


Figure: Sursa figurii: javalearningonline

Mașina Virtuală Java (JVM - Java Virtual Machine)

- Este mediul de rulare (eng. runtime environment) al platformei Java.
- Transformă byte code-ul Java în limbaj mașină care este apoi executat de CPU.
- Include o componentă de compilare la cerere (eng. JIT - Just In Time compiler) care convertește byte code în limbaj mașină nativ.
- Permite (aproape) oricărui program scris în Java să ruleze pe orice calculator pentru care există o mașină virtuală Java (JVM).
- Administrează și optimizează memoria RAM (stivă + heap) cu care lucrează programul.

Ce vom utiliza în acest semestru

- Java Development Kit 17
  Oracle Java
  (https://www.oracle.com/java/technologies/downloads/), **sau**
  Open JDK (https://openjdk.org/projects/jdk/17/)
- IntelliJ IDEA
  (https://www.jetbrains.com/idea/), puteți obține versiunea Ultimate creând cont de student pe https://www.jetbrains.com/shop/eform/students
- Alternativ, puteți utiliza IDE-ul Eclipse (https://www.eclipse.org/downloads/packages/), versiunea Eclipse IDE for Java developers
- Vom folosi platforma GitHub pentru lucrul la laboratoare. Veți avea nevoie de un cont pe https://github.com/, pe care îl puteți utiliza și pentru proiectele voastre.
- Comanda "java -version" vă spune versiunea activă a platformei

O mică comparație I

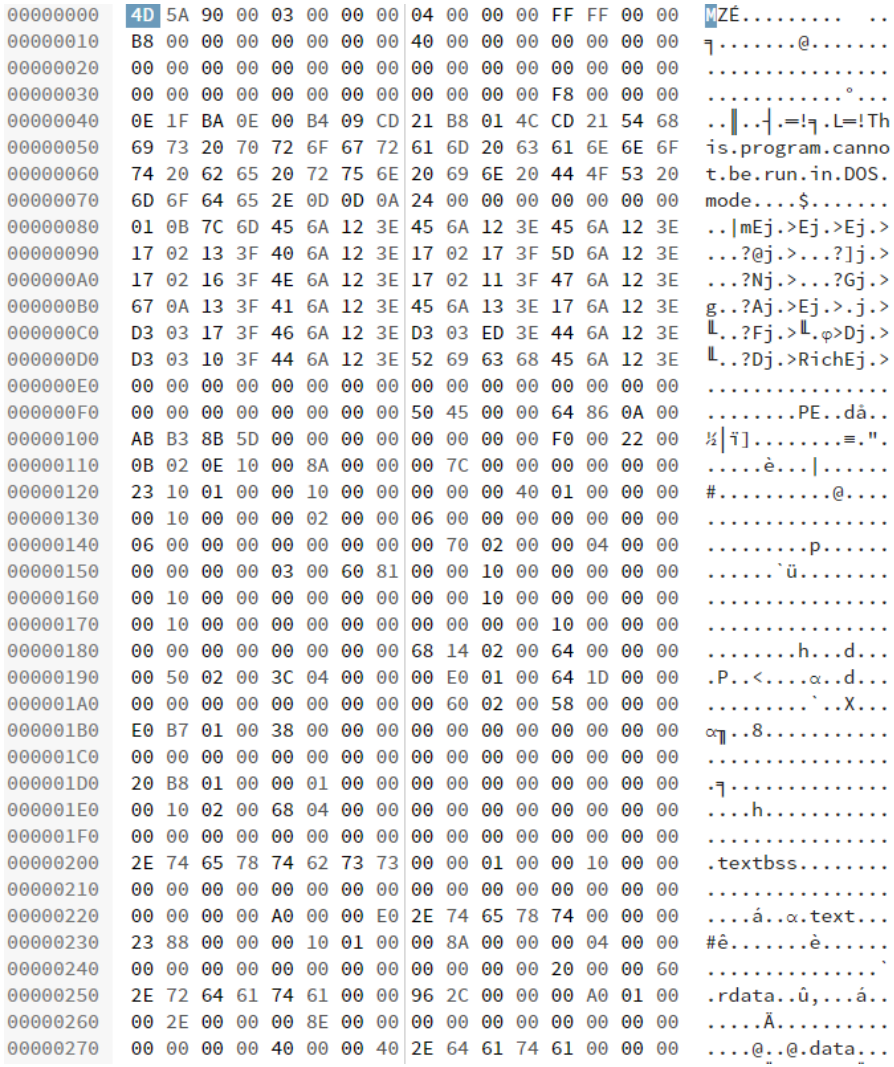
```
#include <iostream>

using namespace std;

int main()
{
    cout << "Hello world!";
    return 0;
}
```

Preprocesare → Compilare → Linkeditare → Executabil (compilat de exemplu pentru arhitectura x86_64, sistemul de operare Windows, executat direct de sistemul de operare)

O mică comparație II



```
00000000  4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00  MZÉ.........  ..
00000010  B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00  ╕.......@.......
00000020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000030  00 00 00 00 00 00 00 00 00 00 00 00 F8 00 00 00  ............°...
00000040  0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68  ..║..┤.=!╕.L=!Th
00000050  69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F  is.program.canno
00000060  74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20  t.be.run.in.DOS.
00000070  6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00  mode....$.......
00000080  01 0B 7C 6D 45 6A 12 3E 45 6A 12 3E 45 6A 12 3E  ..|mEj.>Ej.>Ej.>
00000090  17 02 13 3F 40 6A 12 3E 17 02 17 3F 5D 6A 12 3E  ...?@j.>...?]j.>
000000A0  17 02 16 3F 4E 6A 12 3E 17 02 11 3F 47 6A 12 3E  ...?Nj.>...?Gj.>
000000B0  67 0A 13 3F 41 6A 12 3E 45 6A 13 3E 17 6A 12 3E  g..?Aj.>Ej.>.j.>
000000C0  D3 03 17 3F 46 6A 12 3E D3 03 ED 3E 44 6A 12 3E  ╙..?Fj.>╙.φ>Dj.>
000000D0  D3 03 10 3F 44 6A 12 3E 52 69 63 68 45 6A 12 3E  ╙..?Dj.>RichEj.>
000000E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000000F0  00 00 00 00 00 00 00 00 50 45 00 00 64 86 0A 00  ........PE..då..
00000100  AB B3 8B 5D 00 00 00 00 00 00 00 00 F0 00 22 00  ½│ï]........≡."
00000110  0B 02 0E 10 00 8A 00 00 00 7C 00 00 00 00 00 00  .....è...|......
00000120  23 10 01 00 00 10 00 00 00 00 00 40 01 00 00 00  #..........@....
00000130  00 10 00 00 00 02 00 00 06 00 00 00 00 00 00 00  ................
00000140  06 00 00 00 00 00 00 00 00 70 02 00 00 04 00 00  .........p......
00000150  00 00 00 00 03 00 60 81 00 00 10 00 00 00 00 00  ......`ü........
00000160  00 10 00 00 00 00 00 00 00 00 10 00 00 00 00 00  ................
00000170  00 10 00 00 00 00 00 00 00 00 00 00 10 00 00 00  ................
00000180  00 00 00 00 00 00 00 00 68 14 02 00 64 00 00 00  ........h...d...
00000190  00 50 02 00 3C 04 00 00 00 E0 01 00 64 1D 00 00  .P..<....α..d...
000001A0  00 00 00 00 00 00 00 00 00 60 02 00 58 00 00 00  .........`..X...
000001B0  E0 B7 01 00 38 00 00 00 00 00 00 00 00 00 00 00  α╖..8...........
000001C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000001D0  20 B8 01 00 00 01 00 00 00 00 00 00 00 00 00 00  ·╖..............
000001E0  00 10 02 00 68 04 00 00 00 00 00 00 00 00 00 00  ....h...........
000001F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000200  2E 74 65 78 74 62 73 73 00 00 01 00 00 10 00 00  .textbss........
00000210  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000220  00 00 00 00 A0 00 00 E0 2E 74 65 78 74 00 00 00  ....á..α.text...
00000230  23 88 00 00 00 10 01 00 00 8A 00 00 00 04 00 00  #ê.......è......
00000240  00 00 00 00 00 00 00 00 00 00 00 00 20 00 00 60  ............ ..`
00000250  2E 72 64 61 74 61 00 00 96 2C 00 00 00 A0 01 00  .rdata..û,...á..
00000260  00 2E 00 00 00 8E 00 00 00 00 00 00 00 00 00 00  .....Ä..........
00000270  00 00 00 00 40 00 00 40 2E 64 61 74 61 00 00 00  ....@..@.data...
```

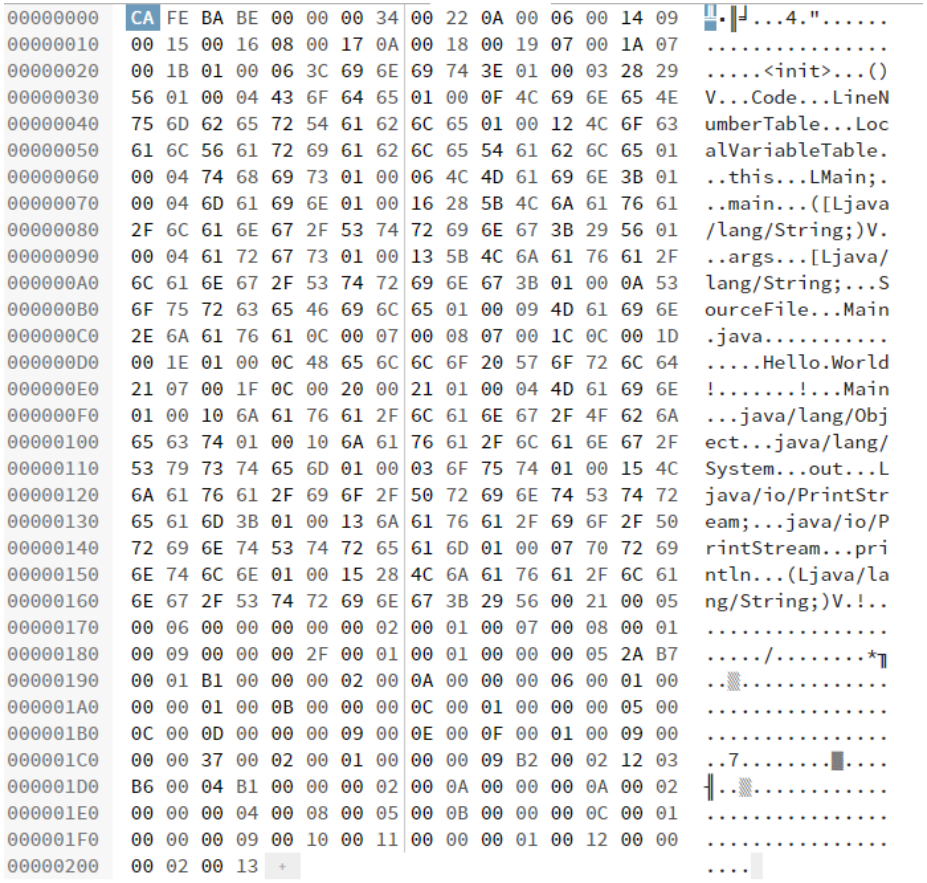
O mică comparație III

```
public class Main {

    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```

Compilare → Byte code (executat de mașina virtuală Java - JVM)

O mică comparație IV



```
00000000  CA FE BA BE 00 00 00 34 00 22 0A 00 06 00 14 09  ▯▪▌...4."......
00000010  00 15 00 16 08 00 17 0A 00 18 00 19 07 00 1A 07  ................
00000020  00 1B 01 00 06 3C 69 6E 69 74 3E 01 00 03 28 29  .....<init>...()
00000030  56 01 00 04 43 6F 64 65 01 00 0F 4C 69 6E 65 4E  V...Code...LineN
00000040  75 6D 62 65 72 54 61 62 6C 65 01 00 12 4C 6F 63  umberTable...Loc
00000050  61 6C 56 61 72 69 61 62 6C 65 54 61 62 6C 65 01  alVariableTable.
00000060  00 04 74 68 69 73 01 00 06 4C 4D 61 69 6E 3B 01  ..this...LMain;.
00000070  00 04 6D 61 69 6E 01 00 16 28 5B 4C 6A 61 76 61  ..main...([Ljava
00000080  2F 6C 61 6E 67 2F 53 74 72 69 6E 67 3B 29 56 01  /lang/String;)V.
00000090  00 04 61 72 67 73 01 00 13 5B 4C 6A 61 76 61 2F  ..args...[Ljava/
000000A0  6C 61 6E 67 2F 53 74 72 69 6E 67 3B 01 00 0A 53  lang/String;...S
000000B0  6F 75 72 63 65 46 69 6C 65 01 00 09 4D 61 69 6E  ourceFile...Main
000000C0  2E 6A 61 76 61 0C 00 07 00 08 07 00 1C 0C 00 1D  .java...........
000000D0  00 1E 01 00 0C 48 65 6C 6C 6F 20 57 6F 72 6C 64  .....Hello.World
000000E0  21 07 00 1F 0C 00 20 00 21 01 00 04 4D 61 69 6E  !.......!...Main
000000F0  01 00 10 6A 61 76 61 2F 6C 61 6E 67 2F 4F 62 6A  ...java/lang/Obj
00000100  65 63 74 01 00 10 6A 61 76 61 2F 6C 61 6E 67 2F  ect...java/lang/
00000110  53 79 73 74 65 6D 01 00 03 6F 75 74 01 00 15 4C  System...out...L
00000120  6A 61 76 61 2F 69 6F 2F 50 72 69 6E 74 53 74 72  java/io/PrintStr
00000130  65 61 6D 3B 01 00 13 6A 61 76 61 2F 69 6F 2F 50  eam;...java/io/P
00000140  72 69 6E 74 53 74 72 65 61 6D 01 00 07 70 72 69  rintStream...pri
00000150  6E 74 6C 6E 01 00 15 28 4C 6A 61 76 61 2F 6C 61  ntln...(Ljava/la
00000160  6E 67 2F 53 74 72 69 6E 67 3B 29 56 00 21 00 05  ng/String;)V.!..
00000170  00 06 00 00 00 00 00 02 00 01 00 07 00 08 00 01  ................
00000180  00 09 00 00 00 2F 00 01 00 01 00 00 00 05 2A B7  ...../........*╖
00000190  00 01 B1 00 00 00 02 00 0A 00 00 00 06 00 01 00  ..▒.............
000001A0  00 00 01 00 0B 00 00 00 0C 00 01 00 00 00 05 00  ................
000001B0  0C 00 0D 00 00 00 09 00 0E 00 0F 00 01 00 09 00  ................
000001C0  00 00 37 00 02 00 01 00 00 00 09 B2 00 02 12 03  ..7........▒....
000001D0  B6 00 04 B1 00 00 00 02 00 0A 00 00 00 0A 00 02  ╢..▒............
000001E0  00 00 00 04 00 08 00 05 00 0B 00 00 00 0C 00 01  ................
000001F0  00 00 00 09 00 10 00 11 00 00 00 01 00 12 00 00  ................
00000200  00 02 00 13                                      ....
```

Cuvântul magic Java

- CAFE BABE (vezi slide anterior)
- James Gosling: *"We used to go to lunch at a place called St Michael's Alley. According to local legend, in the deep dark past, the Grateful Dead used to perform there before they made it big. It was a pretty funky place that was definitely a Grateful Dead Kinda Place. When Jerry died, they even put up a little Buddhist-esque shrine. When we used to go there, we referred to the place as Cafe Dead. Somewhere along the line, it was noticed that this was a HEX number. I was re-vamping some file format code and needed a couple of magic numbers: one for the persistent object file, and one for classes. I used CAFEDEAD for the object file format, and in grepping for 4 character hex words that fit after "CAFE" (it seemed to be a good theme) I hit on BABE and decided to use it. At that time, it didn't seem terribly important or destined to go anywhere but the trash can of history."*

Sintaxa Java

- Derivată din C și C++, asemănătoare cu aceste limbaje.
- Cuvinte cheie (ex. case, for, if, return, void, throws, public, class, boolean, break ... și altele)
- Separatori: ( ) { } ; , .
- Literali: "Hello", 'a', 100, 12.3, true, false, null
- Operatori: *, -, ++, ¡, !=, ——, += ... și alții)
- Comentarii:
  - O singură linie: //
  - Mai multe linii: /* ... */
  - Javadoc: /** ... */

Tipuri de date și referințe

- Tipuri de date primitive: byte, short, int, long, float, double, boolean, și char.
- void - nu reprezintă un tip de dată, nu se poate face cast la void.
- **Referințe**:
  - La creare, se returnează o *referință* la nou obiect.
  - Referințele și tipurile primitive sunt transmise *prin valoare* - limbajul Java transmite parametrii prin valoare
  - Obiectele nu se transmit ca parametri la funcții (nu avem *constructor de copiere* ca în C++), ci referința către obiect se transmite prin valoare.
- null - cuvânt cheie care reprezintă o referință nulă.

Java references vs C++ pointers and C++ references

	Referințe Java	**Pointeri C++**	**Referințe C++**
Referă obiecte	Da	Da	Da
Inițializat cu **new**	Da	Da	Nu
Poate fi **null**	Da	Da	Nu
Poate fi actualizat să refere alt obiect	Da	Da	Nu
Transmis prin valoare	Da	Da	Da
Dereferențiat implicit	Da	Nu	Da

Variabile, constante și instrucțiuni

- Variabile:

```
int x = 10;
String animal = "raccoon";
```

- Constante: final.

```
final int MAX = 32000;
final int MIN;
MIN = -32000;
MIN = 200; // ERROR
```

- Instrucțiuni: if, while, do-while, for, switch.

Tablouri I

- Declarare:

```
type name[]; OR type[] name;
```

- Inițializare:

```
name = new type[DIM];
```

- Toate tablourile în Java sunt alocate dinamic.
- Indexarea tablourilor se face de la 0.
- Dimensiunea unui tablou este returnată de metoda length().

Tablouri II

- Declararea unui tablou N-dimensional:

```
type name[][]...[]; OR type[]...[] name;
```

- Inițializarea:

```
name = new type[DIM_1][DIM_2]...[DIM_n];
```

Șiruri de caractere

- Tablouri de caractere, ex.

```
char[] array = {'a', 'b', 'c'};
```

- Accesarea se face pe bază de index
- Clasa String:

```
String s = "abc";
s += "def";
```

- Compararea șirurilor de caractere (obiecte String): funcția equals(...):

```
String s = new String("abc");
String t = new String("abc");
System.out.println(s.equals(t));
```

Java 101

- Fiecare program Java trebuie să conțiună cel puțin o clasă.
- Numele fișierului Java trebuie să fie același cu numele acestei clase.
- Punctul de intrare în program este o metodă publică, statică numită main().
- Argumentele din linia de comandă ajung în program sub forma parametrului cu care mașina virtuală Java (JVM) va apela funcția main().

Temă

Scrieți un program Java care generează toate numerele prime mai mici ca un număr dat ca parametru la linia de comandă.

Dezvoltarea software I

- Dezvoltarea software este o activitate complexă și provocatoare, care implică un număr mare de activități
- Procesul de dezvoltare software include mai multe faze:
  - Planificarea
  - Analiza
  - Proiectarea
  - Implementarea
  - Testarea și Integrarea
  - Instalarea (eng. deployment)
  - Mentenanța

Dezvoltarea software II

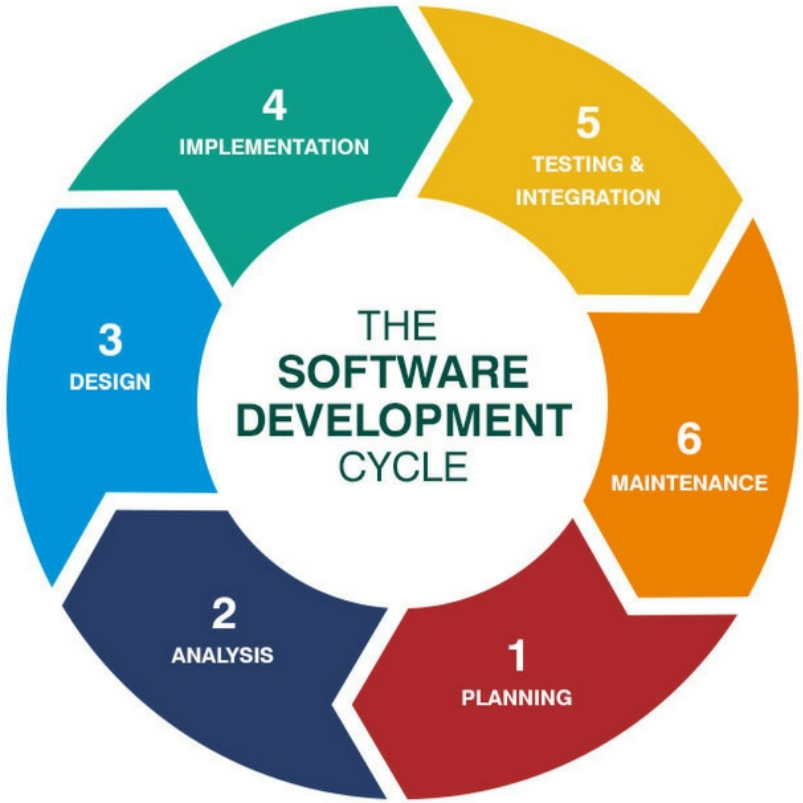


Figure: Sursa f igurii: medium.com

Dezvoltarea software III

- Dificultățile în dezvoltarea software se pot naște din:
  - Modificarea specificațiilor
  - Evoluția tehnologiilor și standardelor
  - Existența echipelor eterogene (posibil distribuite geografic)
  - Nevoie pentru previziuni și estimări de încredere
  - Probleme de comunicare
  - Probleme de integrare

Metodologii de dezvoltare software

- O metodologie de dezvoltare software este un set de reguli, bune practici, valori și principii utilizare în procesul de dezvoltare software.
- Aceste modele furnizează o structură și posibilitatea de a ghida procesul de dezvoltare software.
- Fiecare model își are avantajele și limitările
- Cea mai potrivită metodologie de dezvoltare trebuie aleasă în conformitate cu cerințele proiectului, profilul de risc, costurile asociate, nevoia de predictibilitate, nevoia de a putea demonstra progresul atins, precum și gradul de implicare și feedback oferit de potențialii clienți.

Metodologii Heavyweight

- Metodologia **tradițională** pentru dezvoltarea software.
- Pașii sunt efectuați în ordine secvențială (definirea cerințelor, construirea soluției, testare, implementare)
- Fluxul de dezvoltare este unidirecțional, iar fiecare fază își are livrabilele specifice, bine definite.
- Un set complet de cerințe trebuie să fie definit la demararea proiectului.
- Exemple sunt metodologia **waterfall**, modelul **spiral**, **rational unified process**.

Metodologia Waterfall I

- Unul din cele mai vechi modele, introdus în 1970 de Winston W. Royce (Royce, W.W. (1970) *Managing the Development of Large Software Systems*. Proceedings of IEEE WESCON, 26, 328-388) - rămâne utilizat și astăzi în multe organizații.
- Este un model liniar, în care fiecare fază trebuie încheiată înainte de a demara faza următoare.
- Nu se pretează schimbărilor de cerințe de pot apare în timpul implementării proiectului și nu se bazează pe comunicarea frecventă cu clienții.

Metodologia Waterfall II

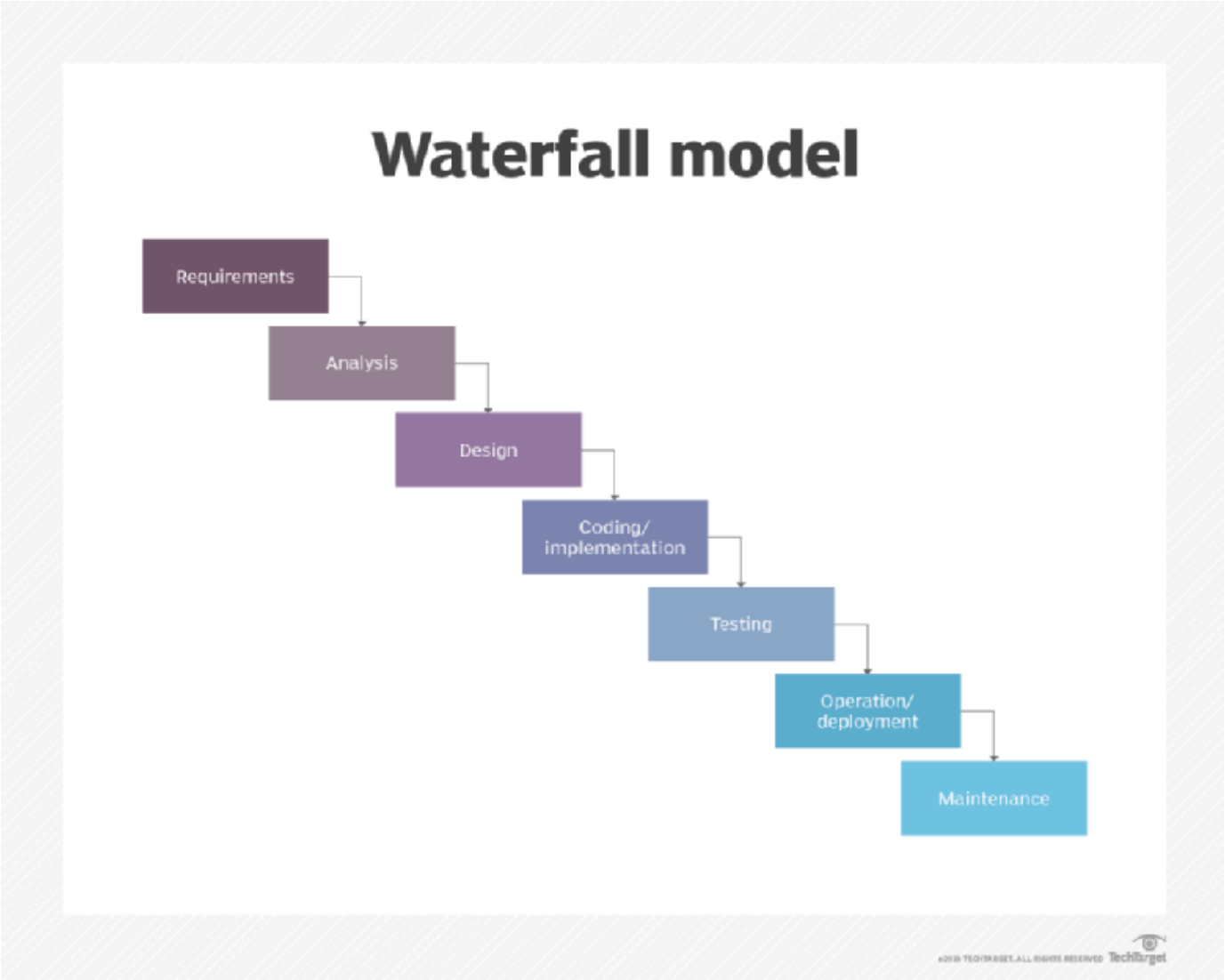


Figure: Figure source: Metodologia Waterfall

Metodologii Agile I

"An iterative and incremental (evolutionary) approach to software development which is performed in a highly collaborative manner by self-organising teams within an effective governance framework with "just enough" ceremony that produces high quality solutions in a cost effective and timely manner which meets the changing needs of its stakeholders."

Sursa: Moniruzzaman, A. B. M., and Hossain, D. S. A. (2013). Comparative study on agile software development methodologies. arXiv preprint arXiv:1307.3356.

Metodologii Agile II

- Abordările Agile se focalizează pe client; aceștia pot interveni propunând schimbări pe durata dezvoltării.
- Cerințele sunt dinamice, pot fi furnizate în mod periodic, pe durata dezvoltării proiectului, ceea ce facilitează consensul dintre clienți și echipa de dezvoltare.
- Clienților li se pot furniza implementări parțial funcționale din timp în timp, air aceștia le vor putea evalua.
- Proiectul e de regulă împărțit în mai multe module care pot fi livrate periodic, și nu neapărat toate în același timp.
- Exemplu sunt metodologia **Scrum**, **Extreme Programming**, **Feature Driven Development**

Metodologii Agile III

Manifest pentru dezvoltarea Agilă

- **Persoane și interacțiuni** $\geq$ Procese și Unelte
- **Software care funcționează** $\geq$ Documentație Exhaustivă
- **Colaborarea cu clienții** $\geq$ Negocierea Contractului
- **Adaptarea la modificări** $\geq$ Respectarea unui plan

Tradițional vs Agil

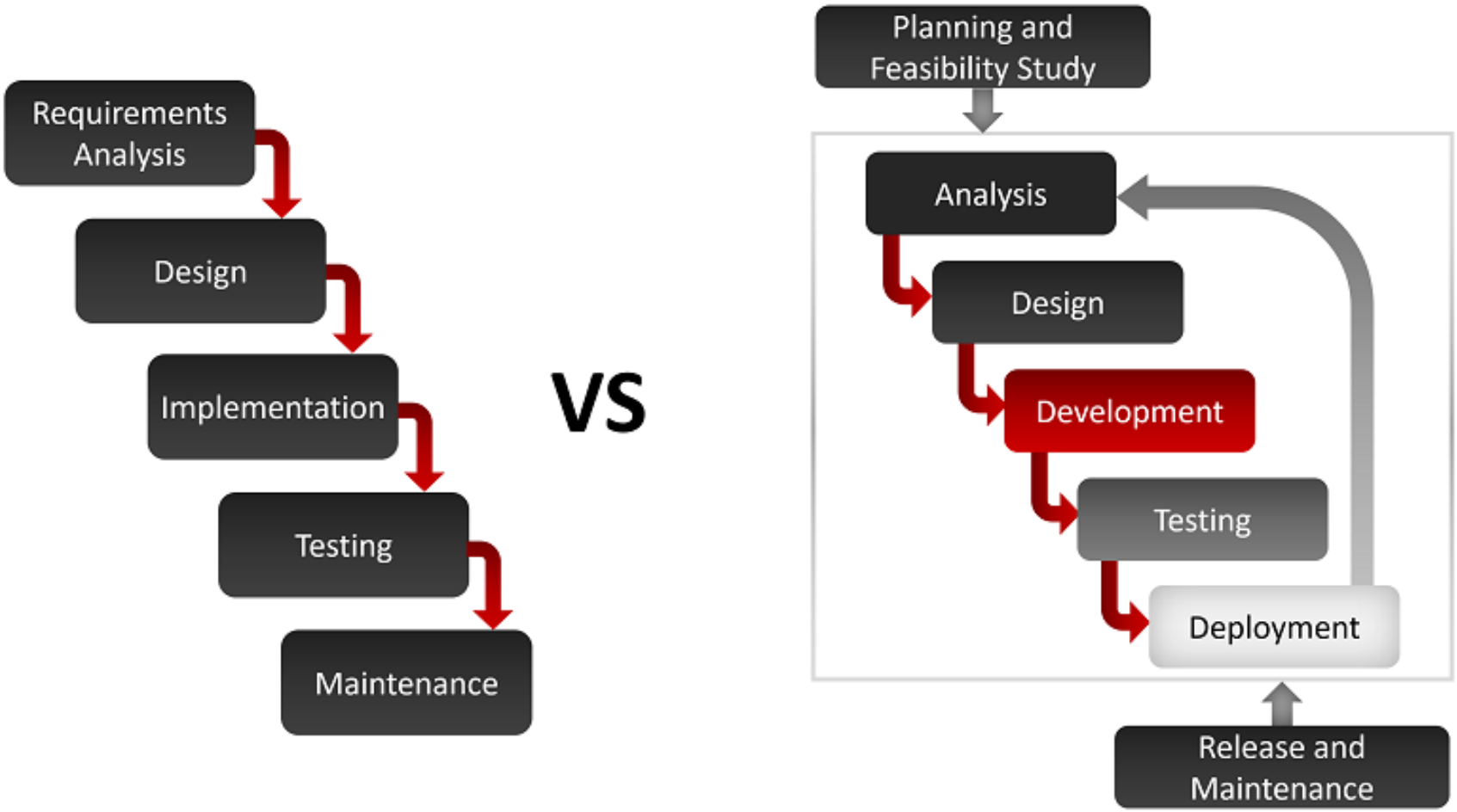


Figure: Figure source: KPI Partners

Tradițional vs Agil

Proprietate	**Tradițional**	**Agil**
Cerințe utilizator	Definite în mod clar înaintea implementării	Dinamic, interactiv, pot fi actualizate
Implicarea clienților	Mică, mai mult la demararea proiectului	Mare, comunicare pe toate durata implementării
Structura organizațională	Liniară	Iterativă
Metodologia de dezvoltare	Ciclul de viață tradițional	Dezvoltarea iterativă, în mod evolutiv
Preferința către metodologie	Favorizează anticiparea	Favorizează adaptarea
Aria de aplicabilitate	Situații în care cerințele sunt bine înțelese de la bun început	Proiecte dinamice

Rezumat

- Limbajul de programare pe care îl vom utiliza în cadrul acestui curs este Java, împreună cu platforma GitHub
- Aveți la dispoziție toate uneltele pentru a vă apuca de treabă
- Procesul de dezvoltare software este complex și include mai multe faze
- Există multe metodologii pentru dezvoltarea software, care se împart în general în două mari categorii: tradiționale și agile