



STEPS TO PROGRAMMING

Урок 1-ви

Въведение и
фундаментални концепции

Цел на курса



- За кого е този курс ?
- Какво ми трябва за курса ?
- Как да разбера дали това е за мен?

За нас

Станислав Иванов

- Софтуерен инженер (BSc Comp. Science)
- Full stack software engineer at Capco
- Java 8, Spring Boot, JavaScript
- React-Native, React, Git



За нас

Даниел Одрински

- Инженер на Компютърни Системи (MEng Hons.)
- Red Hat Certified Systems Administrator (RHCSA)
- Professional Services Engineer @ GoMedia
- Java SE/EE/FX 8 и 17, Go
- Bash, HTML5, CSS3, Docker, Git



Очаквания



Какво мога да очаквам ?



Какво да не очаквам ?

Какво е програмен език?



Дефиниция:

Език четлив за човека състоящ се от директиви, структури и знаци които позволяват на програмиста да изрази поредица от команди (алгоритъм) които да бъдат изпълнени от компютъра като полезна работа.

Какво е „програмиране“?

В програмирането се занимаване с разработката, изграждането и модифицирането на компютърни програми.

Какво е компютърна програма ?

Добре дефинирани, стъпка по стъпка инструкции които компютърът разчита и следва.

В кои сфери се среща програмирането ?

Какви технологии ще използваме ?

- **Java**

- Интерактивни демо програми, които ще бъдат презентирани и до които и вие ще получите достъп, са написани на Java.
- Проект който ще изградим заедно - също написан на Java.

- **Maven**

- **IntelliJ**

- **Git**

Какво е Java?



Java е много популярен програмен език разработен и разпространен от Sun Microsystems през Май 1995 (преди 27 години).

Дефиниция:

Java is a *high-level, class-based, object-oriented programming language* that is designed to have as few implementation dependencies as possible. It is a general-purpose programming language intended to let programmers write once, run anywhere (WORA).

[Wikipedia]

Интересен факт:
Java е тип кафе в
Индонезия.

Изисквания за работа

Образование

Висше образование (поне Бакалавър) – не винаги е задължително ако имаш практичен ОПИТ.

- Трябва да е в сферата на компютрите или инженерството – „A degree in either Computer Science or a technical / engineering discipline“.
- Висше в Математика би помогнало но не е същото като да имаш висше в „Компютърни Науки“.

Сертификати (Certifications) – има изпити които човек може да си плати да изкара за да бъде официално акредитиран като специалист в дадена област или технология.

- Особено могат да помогнат ако нямате висше образование.
- Изпитите могат да струват между £150-£600 (а понякога и повече), в зависимост от престижа на сертификационната организация и нивото на специализацията.
- Bootcamp който се състои в интензивно изучаване на език и концепциите за програмиране което по-принцип трае от 3 до 6 месеца и е обикновено между 20 и 45 часа седмично.

Изисквания за работа

Умения

- Много добро ниво на Английски – разговорен и писмен – задължително.
- Добра комуникация с другите.
- Ефективна работа в екип.
- **Agile development environment** – система за работа при която могат да се приложат няколко различни метода на раздробяване на работата, събрания, планиране и измерване на продуктивността на екипа.

Изисквания за работа

Технологични умения

- **Опит с програмен език (демонстриран) в разработването на софтуер и писане на тестов код.**
- **Algorithms (Алгоритми)** – очаква се един програмист да знае някои от по-често срещаните алгоритми за решаване на сходни и често срещани проблеми. Тук е важно колко стъпки има алгоритъма и за колко време свършва своята работа в сравнение с информацията която му се подава. Някои алгоритми се забавят експоненциално с увеличение на подадената информация, други не се забавят или се забавят с много малко. (Big O notation).
- **Data Collections** – какви структури на информацията съществуват, как работят и подреждат информацията. Тук е важно един програмист да знае коя структура на информацията е подходяща за даден проблем. Различните структури предполагат и различни алгоритми за четене и писане в структурата, което директно афектира скоростта на софтуера който се разработва.

Изисквания за работа

- **Test-driven development (TDD)** – тестов код се пише преди самият код да се напише. Пишем за да угодим на тестовия код и той да мине.
- **Version Control Systems (VCS: Git, Mercurial, SVN)** – системи които помагат за систематична работа с промени в кода чрез версии, предоставяйки синхронизация между хора, екипи и компании.
- **Опит с технологии които се използват в изграждането на системи заедно със софтуера който се разработва:**
 - база данни (relational and non-relational databases/SQL, NoSQL)
 - опашки за съобщения (message queues – MQTT, Kafka)
 - Популярни библиотеки и 'frameworks' които надграждат на езика и се използват от дадената компания. Има някои които се използват от почти всички и за това си заслужава да се научат.

Изисквания за работа

Опит с технологии които се използват за 'пакетиране и инсталация' на софтуера който се разработва, на сървъри – Continuous Integration / Continuous Deployment

- GitLab CI
- Jenkins
- GitHub Actions
- ArgoCD
- Docker
- Kubernetes

Интервю

- Интервютата по принцип се извършват на няколко етапа, всеки от който е с варираща трудност и тема.
- Възможно е първото интервю да е по телефона или на видео обаждање.
- Въпроси които да помогнат на интервюиращите да те опознаят по-добре.
- Технологични въпроси които да изпитат знанията ти на различни теми свързани с технологиите които се използват от компанията.

ИНТЕРВЮ

Code challenge

- Един или няколко проблема за решаване чрез написването на малка програмка или една, две функции/метода.
- По-принцип това се случва на онлайн платформа (Live coding) която проверява кода който се пише чрез 'тестов код' който вкарва различни входни стойности и дали резултатите които излизат от програмата са според очакванията.
- Алтернативна опция в която се дава по-голяма задача и в определен срок зададен от работодателя трябва да се разреши и да им се изпрати решението (в собственото си време).
- Също така, този изпит може да се случи и в присъствието на изпитващи които искат да им обясниш всяка стъпка от мисления си процес и подхода към проблема, както и да го решиш. По някога, изпитващите могат да ти дадат възможността да довършиш задачата въщи и да я изпратиш по-късно.
- Първите няколко интервюта могат да бъдат доста обезкуражителни, но са много полезни когато се приемат като практика и опит.

Какво прави програмиста?

- Разработва нови функционалности в дадена програма според изискванията за проекта.
- Оправя 'бъгове' или грешки/проблеми в кода които водят до грешни или неочаквани резултати.
- Минават през стар код и търсят начини да го направят по-гъвкав, по-изчистен и по-прост така че вероятността кода да има бърк/грешка да е по-ниска, а в същото време, бъдещото надграждане на програмата да е по-лесно и по-систематично структурирано.
- Това понякога може да доведе и до по-добра производителност на програмата, което да означава по-малко загубени клиенти, повече транзакции в секунда, повече пари за компанията.
- Code reviews - Преглеждане на промени в кода които трябва да се обединят със основната база код. Всеки от екипа който преглежда кода може да пише коментари. По-принцип, поне двама трябва да одобряват промените за да бъдат обединени.
- Peer-code sessions
- Срещи с колеги и мениджъри на проекти за да разискват и стигнат до решения които съответстват с изискванията на клиента (проблемите които трябва да се решат). Тук се дискутират архитектура на функцията/програмата/системата/API, начин на работа, условия на операция и употреба.

Какво прави програмиста?

- Работата се раздробява на парчета и всяко парче се записва на ,билети', така че всеки може да поеме по-няколко билета и да извърши работата свързана с билета.
- Среци с колеги и организатори (SCRUM Masters) за да се обсъди работата на екипа, продуктивността, какво върви добре, кое не върви толкова добре и какво може да се подобри, както и да се изгради чувство за принадлежност към екипа и да се дадат възможности на всеки да изкаже благодарност към другите за помощта която са им оказали.
- Разучава APIs и услуги (microservices) с които да интегрира софтуера.
- Постоянно учи.

Заплата

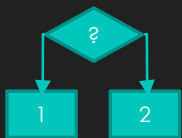
- Започващ, без опит на работно място – около £28,000-£35,000 на година.
- 1-2 години опит – около £35,000-£40,000 на година.
- 2-3 години опит – около £40,000-£50,000 на година.
- 4+ години - £50,000+ на година.

Програма на курса



Първи урок (Въведение)

- I. Въведение в програмиране
- II. Променливи
- III. Прimitives и референтни променливи
- IV. Аритметични оператори
- V. Видове и приложение



Втори урок (Контрол на потока)

- I. Булеви променливи
- II. Булеви оператори
- III. Контрол на потока с if-else statement
- IV. Контрол на потока с switch

Програма на курса



Трети урок (Класове, Обекти и методи)

- I. Класове и обекти
- II. Конструктор и методи
- III. Статични методи и параметри



Четвърти урок (Цикли)

- I. Масиви
- II. Цикъл с for
- III. Цикъл с for each
- IV. Цикъл с while

Програма на курса

Пети урок (Начало на проекта)

- I. Запознаване с кода
- II. Създаване на дата класове
- III. Използване на дата класовете в основният метод

Шести урок (Логически класове)

- I. Създаване на класовете с логическа стойност
- II. Свързване на класовете и методи с останалите логически и дата класовете
- III. Запознаване с визуалните компоненти и техните функции

Програма на курса

Седми урок (Имплементация на функции)

- I. Имплементация на функции и методи
- II. Пренаписване на методи
- III. Запознаване с файл избирателя

Осми урок (Заклучение на курса)

- I. Завършване на проекта
- II. Обобщение на курса
- III. Заклучение на курса

Промелниви и константи

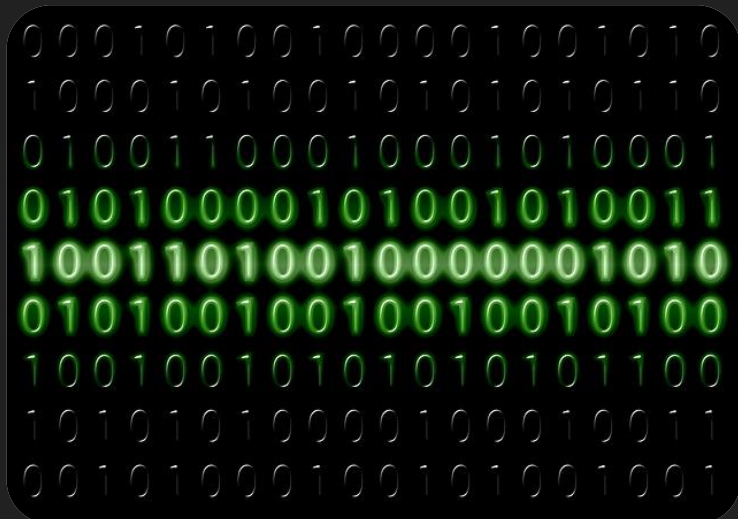
Модификатор, тип и име на променлива	Стойност	Битове
byte <i>age</i>	43	8
int <i>distance</i>	28000	32
bool <i>isHappy</i>	true	1
double <i>height</i>	1.78	64
short <i>days</i>	365	16
long <i>cells</i>	370000000000	64
char <i>letter</i>	i	16
float <i>weight</i>	75.21	32
final bool <i>IS_ALWAYS_HAPPY</i>	true	1

Променливи



- Променливи
- Константи
- Примитивни променливи
- Други видове променливи и константи

Памет



00010100100001001010
10001010010101010110
01001100010001010001
01010000101001010011
10011010010000001010
01010010010010010100
10010010101010101100
10101010000100010011
00101000100101001001

- Къде се съхраняват променливите?
- В каква форма са съхранени?
- Binary system



Демо: Променливи

Аритметични оператори

$+$ $-$ $*$ $/$ $\%$ $=$

$++$ $--$

Няколко вида оператори

- Unary – 1 operand
- Binary – 2 operands
- Ternary – 3 operands

A += 8;

Operands

A = 5 + 10;

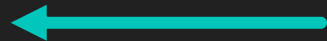
Operator

A = (20 >= 5) ? 10 : 20;

Modulus / Moduli (%)

$$\begin{array}{r} 25 \\ 3 \overline{) 25} \\ \underline{24} \\ 1 \end{array}$$

1



Остатък / Remainder

Положителни и отрицателни числа

+7

-15

Increment operator

```
int cheesecakesEaten = 5;  
System.out.println(cheesecakesEaten++); // Output: 5
```

```
int cheesecakesEaten = 5;  
System.out.println(++cheesecakesEaten); // Output: 6
```

Decrement operator

```
int cheesecakesLeft = 1;  
System.out.println(cheesecakesLeft--); // Output: 1
```

```
int cheesecakesLeft = 1;  
System.out.println(--cheesecakesLeft); // Output: 0 ☹️
```

Слети
оператори
(Compound
assignment
operators)

$A = A + B$

$A = A - B$

$A = A * B$

$A = A / B$

$A = A \% B$

$A += B$

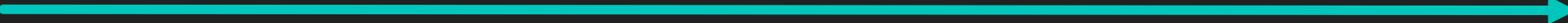
$A -= B$

$A *= B$

$A /= B$

$A \% = B$

Приоритизация на операторите



$*, /, \%$ | $+, -$ | $=$ $+=$ $-=$ $*=$ $/=$ $\%=$

Модификация на приоритизацията на операторите

```
int total = 3 * 5 + 1 * 2 - 4; // total: 13
```



```
int total = 3 * ((5 + 1) * 2 - 4); // total: 24
```

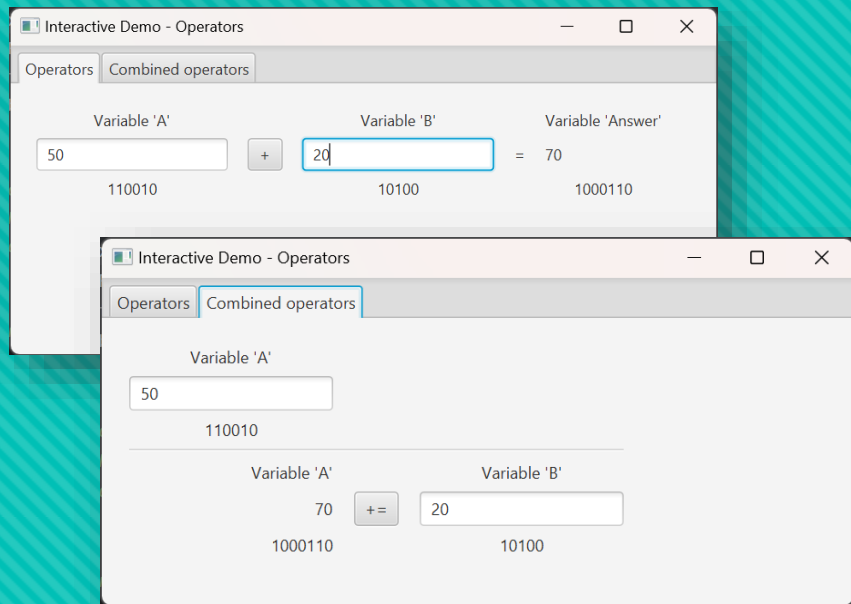
Промоция на числата

Java специфично, 'промотира' типа на променливите на по-големия размер ако извършваме операция с две числа, едно от които използва по-малък тип.

Примери:

- `int + double = double`
- `short + long = long`
- `short + int = int`

Така се гарантира че отговора ще може да се побере в типа си.



Демо: Оператори

Обобщение

- Променливи
- Константи
- Примитивни и не-примитивни видове
- Памет
- Аритметични оператори

Инсталиране на софтуера

Standalone installation

Install IntelliJ IDEA manually to manage the location of every instance and configuration files. For example, if you have a policy that requires specific

Windows Windows (ZIP) macOS Linux

1. Download the installer ↗ .exe.


 There is a separate installer for ARM64 processors.

2. Run the installer and follow the wizard steps.

On the **Installation Options** step, you can configure the following:

On the **Installation Options** step, you can configure the following:

3. Run the installer and follow the wizard steps.

 There is a separate installer for ARM64 processors.

Минимални спецификации на компютъра:

<https://www.jetbrains.com/help/idea/installation-guide.html#requirements>

Инструкции за инсталация (Windows, Linux, macOS):

<https://www.jetbrains.com/help/idea/installation-guide.html#standalone>

Инсталирайте JetBrains IntelliJ IDEA Community Edition (не Ultimate Edition).

Въпроси и Отговори

Тази презентация, ресурси и код:
<https://github.com/stanivanov92/steps-to-programming>