Иван Първанов, Людмил Бонев

ИНФОРМАЦИОННИ ТЕХНОЛОГИИ

10. клас

ПРИТУРКА

Програмиране с JavaScript
уроци 5–8 и тест към раздела

«ДОМИНО» ЕООД

Уважаеми ученици,

Тази притурка е предназначена за паралелките, в които като скриптов език за програмиране се изучава JavaScript. Тя заменя уроци 5–8 от учебника „Информационни технологии за 10. клас“ и съдържа JavaScript варианта на примерите от уроци 4 и 11 и на теста към раздела „Програмиране и изкуствен интелект“. Останалите уроци се изучават по учебника.

Програмите от уроци 5 и 6 се изпълняват направо в уеббраузъра. От урок 7 нататък, когато програмите четат файлове с данни, страницата се отваря чрез локален сървър – например с разширението Live Preview или Live Server за Visual Studio Code.

Файловете за задачите и упражненията към притурката могат да бъдат намерени на адрес **https://kmit.bg**.

Съдържание

Стр.	Тема	Към учебника
3	Към преговора, входния тест и урок 4	с. 6, 7 и 17
4	5. Представяне на алгоритми чрез псевдокод и език за програмиране	вместо с. 20–21
6	6. Първи стъпки с данни в JavaScript	вместо с. 22–23
8	7. Набори от данни: откриване и зареждане	вместо с. 24–25
10	8. Обработка и визуализация на данни (упражнение)	вместо с. 26–27
12	11. Проект: „Анализ на реален набор от данни“	вместо с. 32
13	За любознателните: Papa Parse, Teachable Machine, C++	допълнително
14	JavaScript накратко	справочник
15	Тест: Програмиране и изкуствен интелект	вместо с. 33

Къде пишем и изпълняваме кода

• **Конзолата на браузъра.** Натиснете F12 и изберете раздела Console. Тук се появяват резултатите от `console.log()` и съобщенията за грешки.

• **HTML страница.** Програмата се записва във файл с разширение .js, който се зарежда от HTML страница с реда `<script src="program.js"></script>`. Страницата се отваря в браузъра, а след промяна в кода се презарежда с F5.

• **Node.js (по желание).** Същите програми може да се изпълняват и в терминала с командата `node program.js`.

По притурката не се драска, не се подчертава и не се изписват отговори, тъй като тя ще се използва няколко години. При решаването на задачи и тестове използвайте допълнителен лист.

КЪМ ПРЕГОВОРА И УРОК 4

Преговор: Програмиране и изкуствен интелект (с. 6)

JavaScript е скриптов език за програмиране, който се изпълнява във всеки уеббраузър, а чрез средата Node.js – и самостоятелно на компютъра. Синтаксисът му – фигурни скоби, цикъл `for`, операторите `&&` и `||` – е близък до C++. Възможностите му се разширяват с библиотеки (пакети), които съдържат предварително написан код – от обработка на данни и графики до машинно обучение. Те се съхраняват в общодостъпни хранилища като npm и се добавят в уеб страница с един ред чрез CDN. В професионалната работа с изкуствен интелект водещ език остава Python, но идеите, които изучаваме, са едни и същи и в двата езика.

Входен тест (с. 7): въпроси 3 и 10

3. Какъв резултат ще изведе програмата?

```
let a = 4;
let b = 3;
console.log(a + b * 2);
```

А) 14 Б) 10 В) 11 Г) 24

10. Кое от посочените НЕ е хранилище за съхраняване на библиотеки (пакети)?

А) JavaScript Б) npm В) GitHub Г) GitLab

Урок 4. Алгоритъм (с. 17)

На фиг. 2 е даден пример за алгоритъм за намиране на най-малкото от три числа, реализиран на JavaScript. Сравнете го с псевдокода на фиг. 1 в учебника – всеки ред от псевдокода отговаря на един или няколко реда от програмата.

```
let a = Number(prompt("A ="));
let b = Number(prompt("B ="));
let c = Number(prompt("C ="));
let d = a;
if (b < d) {
  d = b;
}
if (c < d) {
  d = c;
}
console.log("Най-малкото число е:", d);
```

Фиг. 2. Алгоритъм, описан с език за програмиране

Любопитно

Езикът JavaScript е създаден през 1995 г. от Брендън Айк във фирмата Netscape, като първата му версия е написана само за около 10 дни. Въпреки името си той няма общо с езика Java. Днес JavaScript се изпълнява във всеки уеббраузър и е сред най-използваните езици за програмиране в света.

Основни понятия

JavaScript – скриптов език за програмиране, който се изпълнява в уеббраузъра и чрез средата Node.js.

Node.js – среда за изпълнение на JavaScript програми извън браузъра.

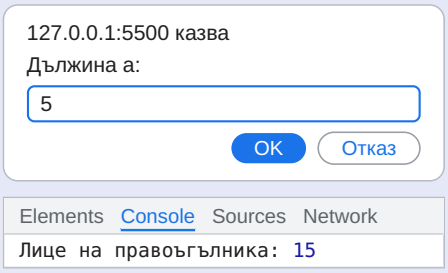
CDN – мрежа от сървъри, от които уеб страниците зареждат готови библиотеки.

5. Представяне на алгоритми чрез псевдокод и език за програмиране

Къде пишем кода

Отворете файла shablon.html в браузъра и натиснете F12 → Console. Там се появяват резултатите от `console.log()`.

Кратък код може да се постави и направо в конзолата. При първото поставяне браузърът може да поиска да напишете allow pasting.



Фиг. 1. Въвеждане с prompt() и резултат в конзолата

Важно!

`prompt()` винаги връща текст. За да смятаме с въведената стойност, я превръщаме в число с `Number()`. Без това `"5" + "3"` дава `"53"`, а не 8.

Дробните числа се въвеждат с точка: 2.5.

Да си припомним основните операции на скриптовия език за програмиране – JavaScript.

Какво прави	Запис на JavaScript
Въвеждане и извеждане на данни	`prompt()` / `console.log()`
Присвояване на стойност	`let x = a * b;`
Проверка на условие	`if (…) { } else if (…) { } else { }`
Повторение определен брой пъти	`for (let i = 1; i <= n; i++) { }`

Задача 1. **Създайте линеен алгоритъм за намиране лице на правоъгълник. Представете алгоритъма чрез псевдокод и с език за програмиране – JavaScript.**

Алгоритъм, описан с псевдокод:

```
Въведи a;
Въведи b;
Пресметни S = a * b;
Изведи S;
```

Алгоритъм, описан с JavaScript:

```
let a = Number(prompt("Дължина a:"));
let b = Number(prompt("Широчина b:"));
let S = a * b;
console.log("Лице на правоъгълника:", S);
```

Задача 2. **Създайте алгоритъм за намиране на най-голямото от три числа, като използвате за a, b, с съответно стойностите 9, 7, 11. Представете алгоритъма чрез псевдокод и с език за програмиране JavaScript.**

Алгоритъм, описан с псевдокод:

```
Въведи a = 9;
Въведи b = 7;
Въведи c = 11;
m = a;
Ако b > m тогава m = b;
Ако c > m тогава m = c;
Изведи m;
```

Алгоритъм, описан с JavaScript:

```
let a = 9;
let b = 7;
let c = 11;
let m = a; // приемаме, че a е най-голямото
if (b > m) {
  m = b;
}
if (c > m) {
  m = c;
}
console.log("Най-голямото е:", m);
```

Важно!

В JavaScript блоковете се ограждат с фигурни скоби `{ }`, а условието на `if` – с кръгли скоби. Отстъпът не е задължителен за компютъра, но е задължителен за човека, който чете кода – затова го спазваме винаги. Всяка нова променлива се обявява с `let`, а командите завършват с `;`.

Задача 3. **Създайте алгоритъм за изчисляване сумата на целите числа от 1 до *n*. Представете алгоритъма чрез псевдокод и с език за програмиране JavaScript.**

Алгоритъм, описан с псевдокод:

```
Въведи n
S = 0
за i от 1 до n със стъпка 1:
    S = S + i
Изведи S
```

Алгоритъм, описан с JavaScript:

```
let n = Number(prompt("Въведете n:"));
let S = 0;
for (let i = 1; i <= n; i++) {
  S = S + i;
}
console.log(S);
```

Записът на цикъла е същият като в C++: начална стойност; условие за продължаване; стъпка (`i++` увеличава `i` с 1).

Проследяване на код (трасиране)

Важно умение е да предвидим резултата на кратък код, без да го изпълняваме.

Задача 4. **Разгледайте дадения алгоритъм. Какъв ще бъде резултатът от изпълнението на кода?**

```
let x = 3;
let y = 4;
if (x + y > 6) {
  console.log("Голямо");
} else {
  console.log("Малко");
}
```

Тъй като `x + y` е 7, а `7 > 6` е вярно, програмата ще изведе **Голямо**.

Задача 5. **Дадена ви е програмата за индекс на телесна маса (BMI) по зададени височина height (в см) и тегло weight (в кг). Стартирайте я за височина 170 см и тегло 65 кг и проверете каква категория извежда.**

```
let height = Number(prompt("Височина (см):"));
let weight = Number(prompt("Тегло (кг):"));
let bmi = weight / (height / 100) ** 2;
if (bmi < 18.5) {
  console.log("Поднормено тегло");
} else if (bmi < 25) {
  console.log("Нормално тегло");
} else if (bmi < 30) {
  console.log("Наднормено тегло");
} else {
  console.log("Затлъстяване");
}
```

За височина 170 см и тегло 65 кг програмата извежда **Нормално тегло**.

Въпроси и задачи

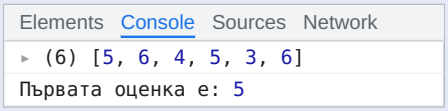
1. На коя конструкция в JavaScript съответства проверката на условие?
2. Каква е ролята на фигурните скоби в JavaScript? Задължителен ли е отстъпът?
3. Какъв ще бъде резултатът от изпълнението на кода от Задача 4, ако x = 10 и y = 1?
4. Създайте код за намиране корените на квадратно уравнение. Коренът се пресмята с `Math.sqrt()`.
5. Представете алгоритъма от Задача 5 чрез псевдокод.

6. Първи стъпки с данни в JavaScript

Основни понятия

Масив (array) – структура от данни, която съхранява няколко стойности в определен ред под едно име.

Индекс – поредният номер на елемент в масива, като броенето започва от 0.



Фиг. 1. Масивът в конзолата

Важно!
Пишете `for (let x of ocenki)` – с `of`. Ако напишете `in`, x получава индексите `"0"`, `"1"`, `"2"`... като текст и вместо сбора на оценките се получава например `"0012"`. Програмата не съобщава за грешка.

Масив – набор от стойности под едно име

Досега работихме с отделни числа. Но истинската сила на програмирането проличава, когато обработваме много стойности наведнъж – например всички оценки на ученик или температурите за един месец. За това в JavaScript използваме **масив**.

Масивът съхранява няколко стойности, подредени една след друга. Записва се в квадратни скоби, а стойностите се разделят със запетая. Достъп до елемент от масива имаме чрез неговия индекс (пореден номер):

```
let ocenki = [5, 6, 4, 5, 3, 6];
console.log(ocenki);
console.log("Първата оценка е:", ocenki[0]);
```

Забележка: *Елементите се броят от 0: `ocenki[0]` е първият елемент, `ocenki[1]` – вторият и т.н.*

Агрегиращи операции

Често не се интересуваме от всяка стойност поотделно, а от обобщен резултат – броя, най-голямата и най-малката стойност, сбора или средната стойност. В JavaScript някои от тях са готови, а сборът се пресмята с цикъл, който вече познаваме от Урок 5.

Запис	Какво връща
`ocenki.length`	броя на елементите
`Math.max(...ocenki)`	най-голямата стойност
`Math.min(...ocenki)`	най-малката стойност
цикъл `for (let x of ocenki)`	сбора – събираме елементите един по един
`ocenki.toSorted((a, b) => a - b)`	нов, подреден масив

Трите точки в `Math.max(...ocenki)` „разгъват“ масива в отделни числа. Цикълът `for (let x of ocenki)` се чете „за всеки елемент x от масива ocenki“.

Задача 1. Създайте алгоритъм за пресмятане на средната стойност като сбор, разделен на броя числа. Разгледайте таблицата с операциите. Използвайте тези, които ви трябват, на мястото на точките.

```
let ocenki = [5, 6, 4, 5, 3, 6];
let sbor = 0;
for (let x of ocenki) {
  sbor = sbor + x;
}
console.log("Брой оценки:", ocenki.length);
console.log("Най-висока:", Math.max(...ocenki));
console.log("Най-ниска:", Math.min(...ocenki));
console.log("Сбор:", sbor);
let sreden = .........;
console.log("Среден успех:", sreden);
```

Попълване на масив от въведени данни

В един масив можем да добавяме нови стойности. За добавяне в края използваме метода `push` (добави). Ето как можем да добавим n стойности, въведени от потребителя:

```
let ocenki = [];
let n = Number(prompt("Колко оценки?"));
for (let i = 0; i < n; i++) {
  let x = Number(prompt("Въведете оценка:"));
  ocenki.push(x);
}
let sbor = 0;
for (let x of ocenki) {
  sbor = sbor + x;
}
console.log("Среден успех:", sbor / ocenki.length);
```

В тези задачи намираме средното аритметично, като разделяме сбора на оценките на техния брой (`ocenki.length`). За изчислението масивът трябва да съдържа поне една стойност – при празен масив JavaScript не съобщава за грешка, а извежда `NaN` („не е число“).

Забележка: *Методът `ocenki.push(x)` добавя стойността на променливата x в края на масива ocenki.*

Задача 2. **Въведете програмата по-горе и я изпробвайте с оценките си по няколко предмета. Разширете я така, че освен средния успех да извежда и най-високата и най-ниската оценка.**

Задача 3. **Създайте масив с дневните температури за една седмица. Изведете средната седмична температура и колко градуса е разликата между най-топлия и най-студения ден.**

Създайте масив със седем стойности. Температурната разлика се получава като `Math.max(...) - Math.min(...)`.

Задача 4. **В час по физическо възпитание учениците бягат 60 метра. Напишете програма, която пита колко ученици са бягали, след което въвежда времето на всеки от тях в секунди и го добавя в масив с push.**

Програмата да извежда: всички времена, подредени от най-доброто към най-слабото, броя на бягалите ученици, най-доброто и най-слабото време и средното време за класа.

Кратък алгоритъм на решението: 1) въвеждане на n; 2) създаване на празен масив `let times = [];` 3) цикъл n пъти – въвеждане на число и добавянето му с `times.push(t);` 4) извеждане на `times.toSorted((a, b) => a - b)`, `Math.min(...times)` и `Math.max(...times)`.

Важно!

Методът `sort()` без функция за сравнение подрежда числата като текст: `[12, 9, 100].sort()` дава `[100, 12, 9]`. Затова за числа пишем `toSorted((a, b) => a - b)` – запомнете го като готов шаблон. `toSorted()` връща нов масив и не променя оригинала.

Задача 5. **Бизнесмен проследява дневните приходи на своя онлайн магазин за една работна седмица (от понеделник до петък). Напишете програма, която въвежда последователно дневните приходи в евро (дробни числа) за петте дни и ги запазва в масив prihodi.**

Програмата да извежда общия оборот за седмицата, среднодневния приход, максималния и минималния приход за един ден. Започнете с празен масив. Използвайте цикъл с пет повторения и добавяйте всяка въведена стойност с `push()`.

Въпроси и задачи

1. Какво е масив и с какво е по-удобен от много отделни променливи?
2. Кой индекс има първият елемент на масив в JavaScript?
3. Как се пресмята средната стойност на масив?
4. Какво прави методът `push()`?
5. Защо `sort()` без функция за сравнение подрежда числата грешно?

7. Набори от данни: откриване и зареждане

Любопитно

Данните се наричат „новият нефт“ на XXI век. Те захранват експертните системи, търсачките и изкуствения интелект. Колкото повече и по-качествени са данните, толкова по-добре работят системите, обучени с тях. Затова умението да намираме и да обработваме данни става все по-ценно.

Зареждане на CSV в Excel

• Изберете менюто Data (Данни) → From Text/CSV (От текст/CSV).
• Маркирайте файла и проверете кодирането (UTF-8) и разделителя.
• Проверете дали данните са разделени в правилните колони.

Основни понятия

Набор от данни (dataset) – организирана съвкупност от данни, обикновено подредени в таблица с редове (записи) и колони (признаци).

Набори от данни

Малките масиви, които въвеждаме на ръка, са удобни за упражнения. В реалния свят обаче се използват големи количества данни, съхранени в структури – наречени набори от данни (dataset). Тяхната обработка е сред основните задачи на съвременните информационни технологии и е в основата на изкуствения интелект.

Откъде да вземем набори от данни?

Съществуват публични хранилища, които предоставят данни безплатно за свободно използване (т.нар. отворени данни):

• Портал за отворени данни на egov.bg – официален сайт с данни за България (население, образование, транспорт и др.).
• Eurostat – статистическа служба на Европейския съюз с данни за всички държави членки.
• Kaggle – световна платформа с хиляди набори от данни по най-различни теми.

Форматът CSV

Най-разпространеният прост формат за таблични данни е CSV (Comma-Separated Values – стойности, разделени със запетая). Това е обикновен текстов файл, в който всеки ред е ред от таблица (запис), а стойностите от колоните са разделени със запетая. Първият ред обикновено съдържа имената на колоните (заглавен ред).

Пример: *Файл **temperaturi.csv** – пет записа с по три стойности (месец, минимална и максимална температура).*

```
месец,мин. температура,макс. температура
Януари,-2,12
Февруари,-4,8
Март,-3,14
Април,2,18
Май,5,28
```

Задача 1. **Разгледайте примерния файл temperaturi.csv. Запишете го с кодиране UTF-8. Поставете го в работната папка на проекта и стартирайте програмата от тази папка. Потърсете в интернет източник на данни за месечните температури във вашия град. Свалете ги като csv файл или ги заменете в съществуващия файл.**

Задача 2. **Посетете Портала за отворени данни egov.bg. Намерете набор от данни по тема, която ви интересува, и разгледайте от какви колони се състои. Запишете: как се казва наборът, колко колони има и какво описва всяка от тях.**

Зареждане на CSV в JavaScript

В JavaScript няма отделен модул за CSV. Файлът се прочита като текст с функцията `fetch()`, след което го разделяме на редове, а всеки ред – на стойности по запетаята. Първо просто прочитаме и извеждаме всеки ред:

```
<!DOCTYPE html>
<html lang="bg">
<head>
  <meta charset="utf-8">
  <title>Урок 7</title>
</head>
<body>
  <script type="module">
    let otgovor = await fetch("temperaturi.csv");
    let tekst = await otgovor.text();
    let redove = tekst.trim().split("\n");
    for (let red of redove) {
      console.log(red.split(","));
    }
  </script>
</body>
</html>
```

Забележка: *Първите три реда в скрипта са готов шаблон за зареждане на файл – използваме ги във всички следващи задачи, като сменяме само името на файла. `fetch()` зарежда файла, а `await` означава „изчакай, докато се зареди“ – затова скриптът е от вида `type="module"`. `trim()` премахва празния ред в края, а `split("\n")` разделя текста на редове.*

По-полезно е да вземем стойностите само от една колона и да ги сложим в масив, за да ги обработим с вече познатите операции. Оттук нататък показваме само съдържанието на `<script type="module">`:

```
let otgovor = await fetch("temperaturi.csv");
let tekst = await otgovor.text();
let redove = tekst.trim().split("\n");

let temperaturi = [];
for (let red of redove.slice(1)) {
  let koloni = red.split(",");
  temperaturi.push(Number(koloni[1]));
}
let sbor = 0;
for (let t of temperaturi) {
  sbor = sbor + t;
}
console.log("Средна минимална температура:",
            sbor / temperaturi.length);
```

Забележка: *`redove.slice(1)` взима всички редове без първия (заглавния); `koloni[1]` е втората колона – минималната температура.*

Важно!

Файлът с данни трябва да е в същата папка като HTML страницата, а страницата – отворена чрез локален сървър (Live Preview или Live Server). При отваряне с двойно щракване браузърът забранява четенето на файла и в конзолата се появява червена грешка. При чужди данни винаги проверявайте първите няколко реда, за да разберете структурата им.

Задача 3. **Създайте текстов файл temperaturi.csv с данните от примера. Въведете втората програма и проверете каква средна температура извежда. Добавете ред, който извежда и най-ниската температура от колоната.**

Задача 4. **Допълнете файла temperaturi.csv със записи за останалите месеци до края на годината. Напишете програма, която зарежда в един масив минималните, а в друг – максималните температури, извежда средната минимална и средната максимална температура и най-голямата месечна амплитуда (разликата между максималната и минималната температура).**

Задача 5. **Създайте файл ocenki.csv, в който всеки ред съдържа учебен предмет и годишната ви оценка по него (заглавен ред: предмет,оценка). Напишете програма, която извежда броя на предметите, средния успех, най-високата и най-ниската оценка, а при среден успех над 5,50 – „Отличен успех!“.**

Ако работите с Node.js

Файлът се чете с един ред, без `await`, а останалият код е същият:

```
let fs = require("fs");
let tekst = fs.readFileSync(
  "temperaturi.csv",
  "utf-8");
```

Въпроси и задачи

1. Какво е набор от данни (dataset)? Дайте пример.
2. Посочете три публични хранилища за отворени данни.
3. Как е устроен файл във формат CSV?
4. Защо в програмата пропускаме заглавния ред със `slice(1)`?
5. Защо страницата с програмата трябва да се отвори чрез локален сървър?

8. Обработка и визуализация на данни (упражнение)

За любознателните

Филтрирането може да се запише и накратко с готовия метод `filter`:

```
let goreshti =
  temperaturi.filter(
    t => t > 20
  );
```

Записът `t => t > 20` е кратка функция: за всяко `t` тя казва дали то да остане в новия масив.

Подбор на данни: филтриране

Често ни трябват само част от данните – например само дните, в които температурата е над определена стойност. Това се нарича филтриране и може да се направи с условие в цикъл.

Задача 1. **Създайте алгоритъм за извеждане броя на дните, в които температурата е над 20 градуса. Разгледайте файла temperatures_july.csv. Той съдържа по един запис за всеки ден със средната дневна температура. Намерете броя на дните със средна температура над 20 °C. По-долу е даден елементарен примерен алгоритъм. За да решите задачата, го съчетайте с програмите от Урок 7.**

```
let temperaturi = [2, 4, 9, 14, 19, 23, 27];
let goreshti = [];
for (let t of temperaturi) {
  if (t > 20) {
    goreshti.push(t);
  }
}
console.log("Дни над 20 градуса:", goreshti.length);
console.log("Стойности:", goreshti);
```

Забележка: *В масива `goreshti` попадат само стойностите, които изпълняват условието `t > 20`.*

Задача 2. **Променете кода така, че да показва броя на дните със средна дневна температура над 25 градуса.**

Подреждане: сортиране

Методът `toSorted()` връща нов, подреден масив. Така лесно намираме крайните стойности и подреждаме данните.

Задача 3. **Създайте алгоритъм за подреждане на стойностите на дните и за намиране стойността на най-студения и най-топлия ден.**

```
let temperaturi = [12, 14, 9, 14, 5, 19, 23, 19, 27];
let podredeni = temperaturi.toSorted((a, b) => a - b);
console.log("Подредени:", podredeni);
console.log("Най-студено:", podredeni[0]);
console.log("Най-топло:", podredeni.at(-1));
```

Забележка: *В JavaScript няма отрицателни индекси: `podredeni[-1]` дава `undefined`. Последният елемент се взима с `podredeni.at(-1)` или с `podredeni[podredeni.length - 1]`.*

```
Elements  Console  Sources  Network
Подредени: (9) [5, 9, 12, 14, 14, 19, 19, 23, 27]
Най-студено: 5
Най-топло: 27
```

Фиг. 1. Резултатът от Задача 3

Визуализация: графика с библиотеката Plotly.js

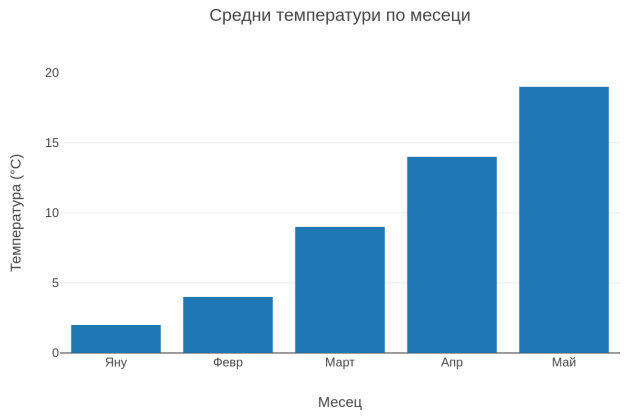
Числата са по-разбираеми, когато ги изобразим графично. За тази цел ще използваме библиотеката Plotly.js. Тя не се инсталира – добавя се в страницата с един ред `<script>`, който я зарежда от интернет (CDN).

Задача 4. **Създайте страница, която чертае стълбовидна диаграма с библиотеката Plotly.js.**

```
<!DOCTYPE html>
<html lang="bg">
<head>
  <meta charset="utf-8">
  <title>Средни температури по месеци</title>
  <script src="https://cdn.jsdelivr.net/npm/plotly.js-dist-min@4.1.2/plotly.min.js"></script>
</head>
<body>
  <div id="grafika"></div>
  <script>
    let mesetsi = ["Яну", "Февр", "Март", "Апр", "Май"];
    let temperaturi = [2, 4, 9, 14, 19];

    Plotly.newPlot("grafika", [{ x: mesetsi, y: temperaturi, type: "bar" }], {
      title: { text: "Средни температури по месеци" },
      xaxis: { title: { text: "Месец" } },
      yaxis: { title: { text: "Температура (°C)" } }
    });
  </script>
</body>
</html>
```

Забележка: *`Plotly.newPlot()` чертае графиката в елемента с `id="grafika"`; `type: "bar"` задава стълбовидна графика. Когато посочите графиката с мишката, над нея се появява лента с бутони – бутонът с фотоапарат я записва като PNG.*



Стълбовидна графика, начертана с Plotly.js

Запомнете

Библиотеките в уеб страница се добавят с ред `<script src="…">` в частта `<head>`.

Ако в конзолата се появи грешка `Plotly is not defined`, проверете този ред и връзката с интернет.

Задача 5. **Заредете набор от данни от папката urok 8 на приложените материали.**

- **Изведете средната, най-високата и най-ниската стойност на една колона.**
- **Филтрирайте записите по избран от вас критерий и пребройте колко са.**
- **Създайте стълбовидна диаграма на предоставените данни с библиотеката Plotly.js. Запазете диаграмата като изображение във файл с име diagram.png, като използвате функцията `Plotly.downloadImage("grafika", { format: "png", filename: "diagram" })`.**

Въпроси и задачи

1. Какво означава „филтриране“ на данни? Как се прави с код?
2. Какво връща методът `toSorted()` и как се взема последният елемент на масив?
3. Коя библиотека използвахме за графики и как се добавя към страницата?

11. Проект: „Анализ на реален набор от данни“

Досега разгледахме как се търси набор от данни, как данните се обработват и как се представят графично с код. Това често се използва в образованието, науката и бизнеса при работа по проекти. Да си припомним, че проектът е организирана работа за постигане на определена цел чрез изпълнение на последователни задачи в определен срок.

Стъпки на проекта

За да реализирате проект, свързан с анализ на набор от данни, трябва да следвате определени стъпки:



1. Определяне на темата и целта.
   - Поставете въпрос – изберете ясен въпрос, на който данните могат да отговорят. Например: „През кой месец е най-топло в моя град?“ или „Кои области в България имат най-много населени места?“.
2. Събиране на данни – откъде ще получим информацията?
   - Изтеглете подходящ набор от публично хранилище (egov.bg, Eurostat, Kaggle) във формат csv.
3. Анализ на данните.
   - Обработете данните – заредете файла с JavaScript (`fetch()` или библиотеката Papa Parse), изчистете ненужното, приложете агрегиращите операции, филтрирайте и сортирайте.
   - Визуализирайте – начертайте подходяща графика с Plotly.js.
4. Изводи – какво показват резултатите?
   - Формулирайте отговора на въпроса с 2 – 3 изречения.
5. Представяне на проекта.
   - Представете проекта пред класа чрез презентация, доклад, постер или кратко устно изложение. Тъй като проектът е уеб страница, може да го покажете направо в браузъра.

Съвет

Може да използвате чатбот с ИИ като помощник – например да обясни съобщение за грешка в конзолата или да предложи как да начертаете графиката. Но винаги проверявайте и разбирайте кода, който получавате – вие отговаряте за резултата, не моделът.

Формула на проекта

За да е постижим, един ученически проект може да използва формулата „1 + 1 + 1 + 1“:

- 1 цел – ясно формулирана (Какво искаме да разберем или да постигнем?).
- 1 набор от данни – от публично хранилище (С какви данни ще работим?).
- 1 анализ – който онагледява отговора (Какво ще анализираме в тези данни?).
- 1 извод – кратко обобщение (До какво заключение стигнахме?).

Проектна задача:

Изберете тема, която ви интересува (спорт, климат, население, образование). Преминете петте стъпки и подгответе кратко представяне (до 3 минути): покажете въпроса, графиката и извода. Може да работите самостоятелно или по двойки. Готов шаблон за проекта с Papa Parse и Plotly.js има на с. 13.

ЗА ЛЮБОЗНАТЕЛНИТЕ

Реални набори от данни с Papa Parse

Ръчното разделяне със `split(",")` се затруднява при реални файлове: стойности в кавички със запетая вътре, разделител `;` вместо запетая, празни редове. Библиотеката Papa Parse решава тези случаи. Добавя се с `<script>` като Plotly.js, а данните се зареждат така:

```
let otgovor = await fetch("danni.csv");
let tekst = await otgovor.text();
let rezultat = Papa.parse(tekst,
                          { header: true, skipEmptyLines: true });
let zapisi = rezultat.data;          // по един запис за всеки ред
let koloni = rezultat.meta.fields;   // имената на колоните

let y = [];
for (let zapis of zapisi) {
  y.push(Number(String(zapis[koloni[2]]).replace(",", ".")));
}
```

Забележка: *Всеки запис съдържа стойностите на един ред, а стойността се взима по името на колоната. `replace(",", ".")` превръща десетичната запетая (12,5) в точка, за да стане число.*

Вашият модел от Teachable Machine в уеб страница

В Урок 9 обучихте модел за разпознаване на чаши и моливи. Teachable Machine може да го превърне в JavaScript код:

- След обучението изберете Export Model → TensorFlow.js.
- Качете модела (Upload) и копирайте готовия JavaScript код от прозореца.
- Поставете кода в HTML страница и я отворете чрез локален сървър.
- Разрешете достъп до камерата и покажете чаша или молив.

Така затваряте кръга „данни → модел → приложение“: моделът, обучен с вашите снимки, работи в собствената ви страница.

От JavaScript към C++

Ако се готвите за състезания по информатика, ще забележите, че много от записите са почти еднакви:

Какво прави	JavaScript	C++
Цикъл с брояч	`for (let i = 1; i <= n; i++)`	`for (int i = 1; i <= n; i++)`
Масив	`let a = [5, 6, 4];`	`vector<int> a = {5, 6, 4};`
Брой елементи	`a.length`	`a.size()`
Добавяне в края	`a.push(x);`	`a.push_back(x);`
Извеждане	`console.log(x);`	`cout << x;`
Подреждане	`a.toSorted((x, y) => x - y)`	`sort(a.begin(), a.end());`

Разлика: `sort` в C++ подрежда самия вектор, а `toSorted` в JavaScript връща нов масив.

Основни понятия

Библиотека – предварително написан код, който добавяме към своята програма.
Запис (обект) – група стойности, всяка с име, например `zapis["месец"]`.

Публикуване на проекта

Готовата страница с графиката може да се публикува в интернет, например в GitHub Pages, и да се сподели с връзка. Преди това се уверете, че данните са отворени и в проекта няма лична информация.

JAVASCRIPT НАКРАТКО

Основни конструкции

Какво прави	Запис	Пример
Нова променлива	`let име = стойност;`	`let S = 0;`
Въвеждане на число	`Number(prompt(…))`	`let n = Number(prompt("n ="));`
Извеждане	`console.log(…)`	`console.log("S =", S);`
Условие	`if (…) { } else { }`	`if (t > 20) { … }`
Сравнение	`=== !== < <= > >=`	`if (D === 0)`
Логически оператори	`&& || !`	`if (a > 0 && b > 0)`
Цикъл с брояч	`for (let i = 1; i <= n; i++)`	`S = S + i;`
Обхождане на масив	`for (let x of a)`	`sbor = sbor + x;`
Брой елементи	`a.length`	`sbor / a.length`
Добавяне в края	`a.push(x)`	`times.push(t);`
Най-голям / най-малък	`Math.max(...a)`	`Math.min(...times)`
Подреждане (числа)	`a.toSorted((x, y) => x - y)`	нов, подреден масив
Последен елемент	`a.at(-1)`	`podredeni.at(-1)`
Без първия елемент	`a.slice(1)`	`redove.slice(1)`
Разделяне на текст	`t.split(",")`	`red.split(",")`
Четене на файл	`await fetch(…)`	`await otgovor.text()`
Корен, степен	`Math.sqrt(x), x ** 2`	`Math.sqrt(D)`
Коментар	`// …`	`// приемаме, че…`

Внимавайте!

- `prompt()` връща текст: `"5" + 3` дава `"53"`. Използвайте `Number()`.
- За обхождане на масив пишете `of`, не `in`: `for (let x of a)`.
- `sort()` без функция подрежда числата като текст. За числа: `toSorted((x, y) => x - y)`.
- `a[-1]` дава `undefined` – последният елемент е `a.at(-1)`.
- В JavaScript няма `sum()` – сборът се пресмята с цикъл.
- Дробните числа се пишат с точка: `Number("2,5")` дава `NaN`.
- Празен масив не дава грешка: `Math.max(...[])` е `-Infinity`, а средното е `NaN`.
- Страница, която чете файлове, се отваря чрез локален сървър, не с двойно щракване.

Как да разчетем грешка

Грешките се изписват в червено в конзолата (F12 → Console), заедно с файла и реда, на който са възникнали.

- `… is not defined` – грешно изписано име или незаредена библиотека.
- `Unexpected token` – липсва или е излишна скоба, кавичка или запетая.
- `Failed to fetch` или CORS – страницата не е отворена чрез локален сървър.
- `NaN` в резултата – текст вместо число или празен масив.
- `undefined` – несъществуващ индекс или свойство.

ТЕСТ

1. На коя конструкция в JavaScript съответства логическият блок (ромб) от блок-схемата?

А) цикъл `for`
Б) условие `if … else`
В) функцията `console.log()`
Г) масив

2. Какъв ще бъде резултатът от изпълнението на кода:

```
let S = 0;
for (let i = 1; i < 4; i++) {
  S = S + i;
}
console.log(S);
```

А) 3 Б) 6 В) 4 Г) 10

3. Какъв е индексът на първия елемент на масив в JavaScript?

А) 1 Б) 0 В) -1 Г) зависи от дължината

4. Как се получава броят на елементите в масива a?

А) `a.push()` Б) `Math.max(...a)` В) `a.length` Г) `a.toSorted()`

5. Променливата sbor съдържа сбора на елементите на непразния масив ocenki. Как се пресмята средната стойност?

А) `Math.max(...ocenki) / 2`
Б) `sbor / ocenki.length`
В) `ocenki.length / sbor`
Г) `ocenki.toSorted()`

6. Какво представлява набор от данни (dataset)?

А) единична променлива
Б) организирана съвкупност от данни
В) програма за чертане на графики
Г) вид сензор

7. Файлът във формат CSV е:

А) изпълнима програма
Б) текстов файл със стойности, разделени със запетая
В) некомпресирано изображение
Г) файл с аудио- и видеоданни

8. С коя част от данните проверяваме обучения модел?

А) с тренировъчния набор
Б) с тестовия набор
В) с целия набор наведнъж
Г) моделът не се проверява

9. Преобучаване (overfitting) означава, че моделът:

А) работи добре с нови данни
Б) се е приспособил твърде точно към конкретни данни
В) няма достатъчно данни и е склонен към халюциниране
Г) е без сензори

10. Кое НЕ е правило за добра заявка към ИИ?

А) яснота
Б) специфичност
В) въвеждане на лични пароли
Г) посочване на желан формат

11. Защо казваме, че ИИ работи като „черна кутия“?

А) защото е черен на цвят
Б) защото не виждаме как точно стига до отговора си
В) защото няма нужда от данни
Г) защото работи само офлайн

12. Кой изучен инструмент е предназначен за диаграми и графики на данни?

А) `fetch()` Б) `Math.random()` В) Plotly.js Г) `console.log()`

Използвани източници:

Документация на езика JavaScript (MDN Web Docs)
https://developer.mozilla.org

Библиотека Plotly.js
https://plotly.com/javascript

Библиотека Papa Parse
https://www.papaparse.com

Среда Node.js
https://nodejs.org

Teachable Machine
https://teachablemachine.withgoogle.com

Информационни технологии
10. клас
Притурка: програмиране с JavaScript

Към учебника „Информационни технологии за 10. клас“
на Иван Първанов и Людмил Бонев (ISBN 978-954-651-309-0)

Работен вариант, 2026 г.

Издателство Домино ЕООД
6000 Стара Загора, пл. „Берое“ 5
тел./факс 042265173, 042265183
e-mail: office@domino.bg
www.domino.bg