1. **BASIC SYNTAX, CONDITIONAL STATEMENTS AND LOOPS**
   1. **Четене от конзолата**

int age = Integer.parseInt(scanner.nextLine()); //стринг с преобразуване към инт

String username = scanner.nextLine(); // стринг

double pricePerCapsule = Double.parseDouble(scanner.nextLine()); // стринг към дабъл

* 1. **Преобразуване между типове данни**

Кастване

char symbol = 'A';

int asciiValue1 = (int) symbol; // explicit casting, символ към аски код

int asciiValue2 = symbol; // implicit casting

String number = "145";

char symbol = number.charAt(0); //от стринг към символ

String symbolAsText = symbol + ""; // от символ към стринг

int digit = Integer.parseInt(symbolAsText); // от стринг към int

int digit = Integer.parseInt(inputNumber.charAt(position) + "") //позиция от стринг към инт

* 1. **Форматирано отпечатване**

System.out.printf("Name: %s, Age: %d, Grade: %.2f", name, age, averageGrade);

\n – нов ред

* 1. **Други**

group.equals("Students") // стрингът group е еднакъв на Students

!input.equals("Start") // стрингът input не е еднакъв със стрингът start

number % 2 != 0 // числото не е четно, %2 – остатък при делене на две

int lastDigit = number % 10 // последна цифра на число

number /= 10; // премахване на последната цифра на число

if (Character.isUpperCase(symbol) // проверка дали стринг започва с главна буква

1. **МАСИВИ**
   1. **Обща информация**

//1. масивът е съвкупност от еднотипни елементи

//2. масивът има постоянна дължина array.length

//3. дължина на масив = максималния брой елементи, които можем да съхраним

//4. позиции / индекси -> 0 до последната (дължина на масива - 1)

//5. задавате стойност в масив: array[0] = 56;

//6. достъпвате стойност в масив: array[5]

* 1. **Създаване на масив**

String[] cars = {"Volvo", "BMW", "Ford", "Mazda"};

* 1. **Пълнене на масив**

//вариант 1 за запълване на масив(статичен с предварително зададени елементи)

int [] dates = {4, 5, 6, 7};

//вариант 2 за запълване на масив (празен масив и добавяме елементи)

double[] prices = new double[10];

prices[0] = 34.5;

prices[1] = 23.5;

//вариант 3 за запълване на масив (елементите се въвеждат от конзолата на отделни редове)

int n = Integer.parseInt(scanner.nextLine());

int [] numbers = new int[n];

for (int position = 0; position <= numbers.length - 1; position++) { numbers[position] = Integer.parseInt(scanner.nextLine());

//вариант 4 за запълване на масив (елементите са на един ред)

String[] inputs = scanner.nextLine().split(" ");

int [] integerNumbers = Arrays.stream(scanner.nextLine().split(" ")).mapToInt(Integer::parseInt).toArray(); //int

double [] decimalNumbers = Аrrays.stream(scanner.nextLine().split(")).mapToDouble(Double::parseDouble).toArray(); //double

String names = "Desi Ivan Georgi Tanya";

String [] namesArray = names.split(" "); //["Desi", "Ivan", "Georgi", "Tanya"] //string

* 1. **Обединяване и преобразуване на елементите на масив**

System.out.println(String.join(" ", namesArray)); //"Desi Ivan Georgi Tanya" // обединяване

System.out.println(Arrays.toString(numbers).replace("[", "").replace("]", "") // Обединяване със замяна на разделител

"Desi".toCharArray() -> ['D', 'e', 's', 'i'] // Стринг към символен масив

* 1. **Извеждане на елементите на масив**

//използваме за обхождане for: има значение позицията на елемента

for (int position = 0; position <= daysOfWeek.length - 1; position++) {

System.out.println(daysOfWeek[position]);

}

//използваме за обхождане foreach: няма значение позицията на елемента

["Monday", "Tuesday", "Wednesday", "Thursday","Friday", "Saturday", "Sunday"]

for (String day : daysOfWeek) {

System.out.println(day);

}

//отпечтване в обратен ред

//[10, 20, 30]

//обратен ред: последната позиция към първата

for (int position = numbers.length - 1; position >= 0; position--) {System.out.print(numbers[position] + " ");}

1. **СПИСЪК - LIST**

**1. Размер на списък**

System.out.println(numbers.size()); //размер на списъка

**2. Добавяне на елемент към списъка**

numbers.add(50); //добавя елемента в края на списъка -> {50}

numbers.add(0, 23); //вмъкваме елемент на дадена позиция; изместваме останалите

numbers.set(1, 45); //заменя елемента на дадена позиция с дадения елемент

**3. Премахване на елемент от списъка**

numbers.remove(Integer.valueOf(50)); //премахва първото срещане дадения елемент от списъка

numbers.remove(1); //премахва елемента на дадената позиция

**4. Отпечатваме списък**

**for цикъл -> трябва да работим с позициите**

for (int position = 0; position <= numbers.size() - 1; position++) {

System.out.println(numbers.get(position));

}

**foreach -> трябва да работим само с елементите**

for (int number : numbers) {

System.out.println(number);

}

**toString**

//numbers = {3, 5, 6, 12,34, 56};

System.out.println(numbers.toString());

//[3, 5, 6, 12, 34, 56]

**String.join -> само за лист от текстове**

List<String> names = new ArrayList<>(Arrays.asList("Ivan", "Georgi", "Pesho"));

System.out.println(String.join(", ", names));

**5. Други методи на list**

System.out.println(numbers.contains(12)); //проверява дали даден елемент е в листа

System.out.println(numbers.isEmpty()); //проверява дали листът е празен

numbers.clear(); //премахва всички елементи в листа

System.out.println(numbers.indexOf(56)); //Позицията, на която се намира елемента; -1 ако няма такъв елемент

size() – number of elements in the List<E> //извежда броя на елементите

add(element) – adds an element to the List<E> //добавя елемент в края

add(index, element) – inserts an element to given position //добавя елемент на определен индекс

remove(element) – removes an element (returns true / false) //премахва даден елемент

remove(index) – removes element at index // премахва елемент на определен индекс

contains(element) – determines whether an element is in the list //проверява дали списъкът съдържа определен елемент

set(index, item) – replaces the element at the given index //заменя елемент на определен индекс

**6. Четене на лист от конзолата**

//34 56 12 45 87

Scanner scanner = new Scanner(System.in);

int[] numbersArray = Arrays.stream(scanner.nextLine().split(" "))

.mapToInt(Integer::parseInt).toArray();

//"34 56 12 45 87".split(" ") -> ["34", "56", "12", "45", "87"] -> [34, 56, 12, 45, 87] -> {34, 56, 12, 45, 87}

List<Integer> numbersList = Arrays.stream(scanner.nextLine().split(" "))

.map(Integer::parseInt).collect(Collectors.toList());

//"Desi Ivan Georgi Pesho".split(" ") -> ["Desi", "Ivan", "Georgi", "Pesho"] -> {"Desi", "Ivan", "Georgi", "Pesho"}

List<String> namesList = Arrays.stream(scanner.nextLine().split(" ")).collect(Collectors.toList());

**7. Сортиране**

7.1. ascending order -> нарастващ ред

Collections.sort(numbers);

7.1. descending order -> намаляващ ред

Collections.sort(numbers); //ascending order

Collections.reverse(numbers); //обратен редДруги методи

**8. Всички методи на ЛИСТ**

|  |  |
| --- | --- |
| void add(int index, E element) | It is used to insert the specified element at the specified position in a list. |
| boolean add(E e) | It is used to append the specified element at the end of a list. |
| boolean addAll(Collection<? extends E> c) | It is used to append all of the elements in the specified collection to the end of a list. |
| boolean addAll(int index, Collection<? extends E> c) | It is used to append all the elements in the specified collection, starting at the specified position of the list. |
| void clear() | It is used to remove all of the elements from this list. |
|  |  |
| boolean equals(Object o) | It is used to compare the specified object with the elements of a list. |
| int hashcode() | It is used to return the hash code value for a list. |
| E get(int index) | It is used to fetch the element from the particular position of the list. |
| boolean isEmpty() | It returns true if the list is empty, otherwise false. |
| int lastIndexOf(Object o) | It is used to return the index in this list of the last occurrence of the specified element, or -1 if the list does not contain this element. |
| Object[] toArray() | It is used to return an array containing all of the elements in this list in the correct order. |
| <T> T[] toArray(T[] a) | It is used to return an array containing all of the elements in this list in the correct order. |
| boolean contains(Object o) | It returns true if the list contains the specified element |
| boolean containsAll(Collection<?> c) | It returns true if the list contains all the specified element |
| int indexOf(Object o) | It is used to return the index in this list of the first occurrence of the specified element, or -1 if the List does not contain this element. |
| E remove(int index) | It is used to remove the element present at the specified position in the list. |  |
| boolean remove(Object o) | It is used to remove the first occurrence of the specified element. |  |
| boolean removeAll(Collection<?> c) | It is used to remove all the elements from the list. |  |
| void replaceAll(UnaryOperator<E> operator) | It is used to replace all the elements from the list with the specified element. |  |
| void retainAll(Collection<?> c) | It is used to retain all the elements in the list that are present in the specified collection. |  |
| E set(int index, E element) | It is used to replace the specified element in the list, present at the specified position. |  |
| void sort(Comparator<? super E> c) | It is used to sort the elements of the list on the basis of specified comparator. |  |
| Spliterator<E> spliterator() | It is used to create spliterator over the elements in a list. |  |
| List<E> subList(int fromIndex, int toIndex) | It is used to fetch all the elements lies within the given range. |  |
| int size() | It is used to return the number of elements present in the list. |  |

**9. Сравнение на ЛИСТ и МАСИВ**

**1. Съзадаване**

int [] array = new int[10]; //задаваме брой на елементите

List<Integer> list = new ArrayList<>(); //няма нужда да задаваме брой елементи

**2. Брой елементи**

System.out.println(array.length); //дължина = брой елементи

System.out.println(list.size()); //размер = брой елементи

**3. Достъп елементи**

System.out.println(array[0]);

System.out.println(list.get(0));

**4. Добавяме елементи**

array[1] = 5;

list.add(50); //добавяме елемента в скобите в края на списъка

list.add(0, 12); //вмъквате елемента на дадения индекс

**5. Обхождане с foreach**

for (int number: array) {

System.out.println(number);

}

for (int number: list) {

System.out.println(number);

}

**6. Обхождане с for**

for (int position = 0; position <= array.length - 1; position++) {

System.out.println(array[position]);

}

for (int position = 0; position <= list.size() - 1; position++) {

System.out.println(list.get(position));

}

//ПРЕДИМСТВА НА ЛИСТ ПРЕД МАСИВ

//1. ПРЕОРАЗМЕРЯВАНЕ - НЯМА НУЖДА ПРЕДВАРИТЕЛНО ДА ЗНАЕМ БРОЯ НА ЕЛЕМЕНТИ

//2. ПО-ФУНКЦИОНАЛЕН = ПО-ЛЕСНО СЕ МОДИФИЦИРА

1. **МЕТОДИ**

**Определение:**

Именуван блок от код, който може да бъде използван по-късно

**Дефиниране (деклариране)**

1. Без параметри

public static void name () {

System.out.println("Hello!");

}

1. С параметри

static void printNumbers(int start, int end) {

for (int i = start; i <= end; i++) {

System.out.printf("%d ", i);

}

}

**Използване**

printHello (); - с параметри

printNumbers (6, 12); - без параметри

**Видове**: връщат стойност и не връщат стойност (void),

**Overloading**: когато няколко метода са с еднакво име, но с различна сигнатура (входни параметри). Всеки метод можа да връща различен тип данни.

1. **ОБЕКТИ И КЛАСОВЕ**
2. **АСОЦИАТИВНИ МАСИВИ**
3. **ПОБИТОВИ ОПЕРАЦИИ**
4. **ОБРАБОТКА НА ТЕКСТОВЕ**