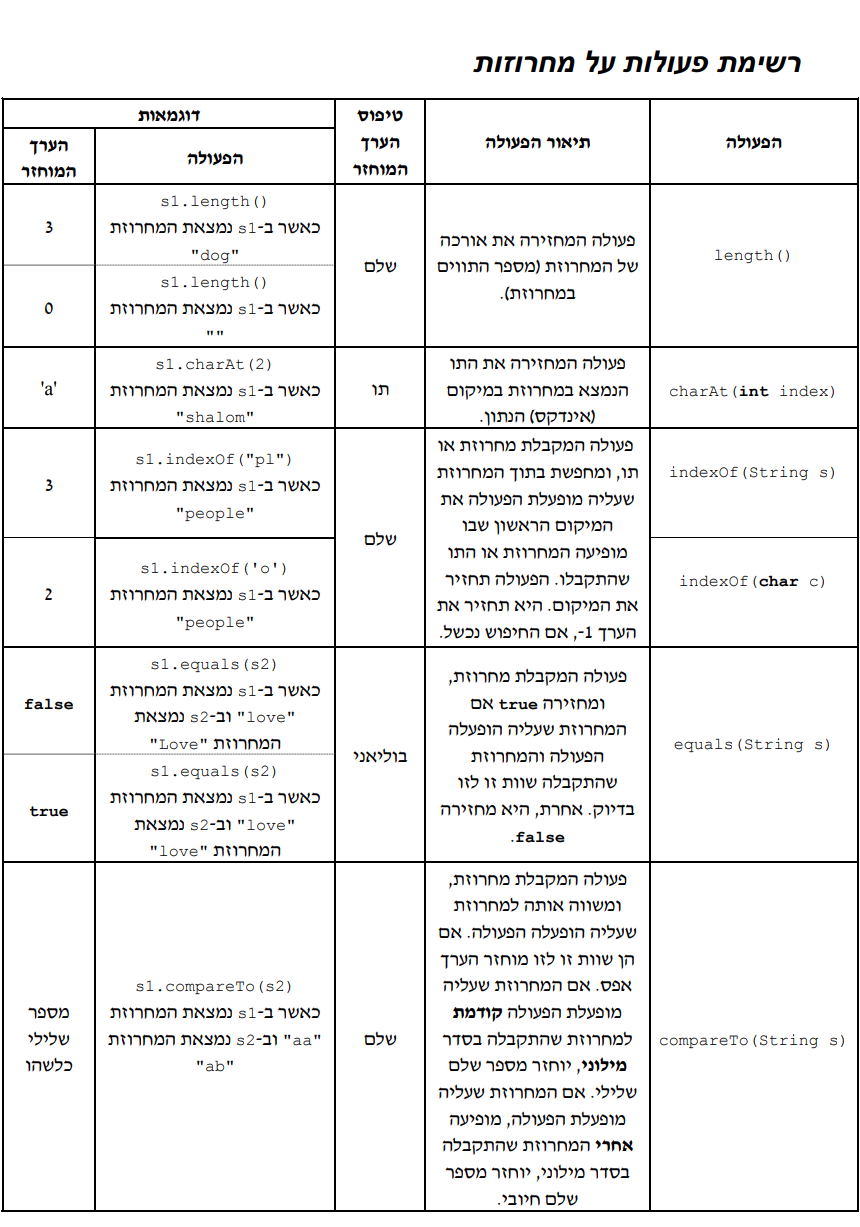
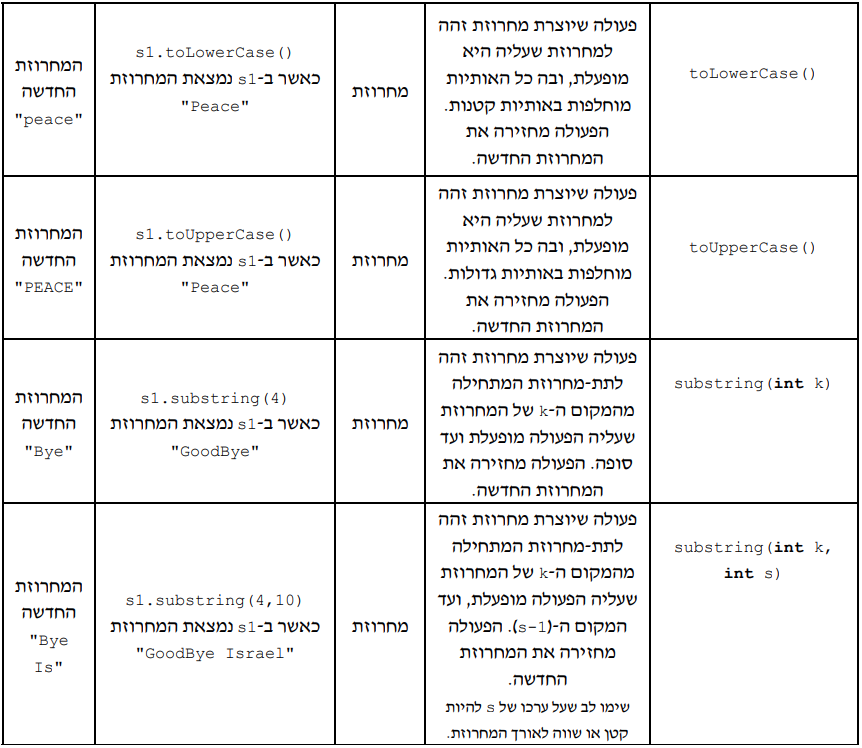
**סיכום מחרוזות ומערכים**

**מחרוזות:**





**קליטת מחרוזת או תו:**

str = in.next(); – קליטת מילה אחת

str = in.nextLine(); - קליטת שורה שלמה

letter = in.next().charAt(0); - קליטת תו

* הצהרה על מחרוזת: **String Str = "example";** (בשונה מטיפוס רגיל, הצהרה על טיפוס מחרוזת מחוייבת להתחיל באות גדולה. **חובה להשתמש במרכאות כפולות** כאשר מבצעים השמה בתוך משתנה מסוג מחרוזת, **לא ניתן להשתמש במרכאות בודדות**).
* הצהרה על תו: **char letter = 'e';** (**חובה להשתמש במרכאות בודדות** כאשר מבצעים השמה בתוך משתנה מסוג תו, **לא ניתן להשתמש במרכאות כפולות**).

**מערכים:**

הצהרה על מערך: הוספת [ ] לפני הגדרת הטיפוס, לדוגמא: **int[] array = new int[10]**.

קבלת אורך המערך: **array.length();**.

פונקציה להדפסת מערך:

public static void printArray(**type[]** arr) { // נדרש לשנות את הטיפוס בטענת הכניסה

for(int i = 0; i < arr.length(); i++)

System.out.print(arr[i] + " ");

}

**סוגי מיון מערכים:**

**מיון בועות:**

public static void bubbleSort(int[] arr) { // BubbleSort

boolean sorted = false;

for (int len = arr.length; len > 0 && !sorted; len--) {

sorted = true;

for (int i = 0; i < len - 1; i++)

if (arr[i] > arr[i + 1]) {

int temp = arr[i];

arr[i] = arr[i + 1];

arr[i + 1] = temp;

sorted = false;

}

}

}

**מיון בחירה:**

public static void selectionSort(int arr[]) { // SelectionSort

int maxIdx, temp;

for (int i = arr.length; i > 0; i--) {

maxIdx = 0;

for (int j = 0; j < i; j++)

if (arr[j] > arr[maxIdx])

maxIdx = j;

temp = arr[i - 1];

arr[i - 1] = arr[maxIdx];

arr[maxIdx] = temp;

}

}

**חיפוש ליניארי:**

public static int linearSearch(int[] arr, int num) { // LinearSearch

for (int i = 0; i < arr.length; i++)

if (arr[i] == num)

return i;

return -1;

}

**חיפוש בינארי:**

public static int binarySearch(int[] arr, int num) { // BinarySearch

int low = 0, high = arr.length - 1;

while (low <= high) {

int mid = (high + low) / 2;

if (arr[mid] == num)

return mid;

if (arr[mid] < num)

low = mid + 1;

else

high = mid - 1;

}

return -1;

}

**בדיקה האם המערך מסודר בסדר עולה או יורד:**

public static boolean ifEqual(int[] arr) {

for (int i = 0; i < arr.length; i++) {

if (i == arr.length - 1)

break;

if (arr[i] **>** arr[i + 1]) // לשינוי הבדיקה בסדר יורד יש להפוך את הסימן

return false;

}

return true;

}

**בדיקה מהו התו הפופולרי:**

public static char mostPopularChar(char[] chars) {

int i = 0, count = 0, maxCount = 0;

char popular = chars[0];

while (i < chars.length) {

int j = 0;

while (j < chars.length) {

if (chars[i] != chars[j] && maxCount < count) {

maxCount = count;

count = 0;

popular = chars[i];

}

if (chars[i] == chars[j])

count ++;

else

count = 0;

j++;

}

count = 0;

i++;

}

return popular;

}

**מציאת חיתוך בין 2 מערכים:**

public static void findIntersection(char[] arr1, char[] arr2) {

int counter = 0;

for (int i = 0; i < arr1.length; i++)

for (int j = 0; j < arr2.length; j++)

if (arr1[i] == arr2[j])

counter++;

char[] arr3 = new char[counter];

int k = 0;

for (int i = 0; i < arr1.length; i++)

for (int j = 0; j < arr2.length; j++)

if (arr1[i] == arr2[j]) {

arr3[k] = arr1[i];

k++;

}

for (int i = 0; i < arr3.length; i++)

System.out.print(arr3[i] + " ");

}

**אופרטורים בג'אווה:**

a = 5 = 0101 (בבינארי)

b = 7 = 0111 (בבינארי)

**| אופרטור OR** בין 2 INT ,כאשר כל INT הינו ביט

0101

| 0111

\_\_\_\_\_\_\_\_

0111 = 7 (In decimal)

a = 5 = 0101 (In Binary)

b = 7 = 0111 (In Binary)

**Bitwise & AND** Operation of 5 and 7

0101

& 0111

\_\_\_\_\_\_\_\_

0101 = 5 (In decimal)

a = 5 = 0101 (In Binary)

b = 7 = 0111 (In Binary)

**Bitwise ^ XOR** Operation of 5 and 7

0101

^ 0111

\_\_\_\_\_\_\_\_

0010 = 2 (In decimal)