

HashSet: Time Complexity

	HashSet	Array	Sorted Array
Insert / Add :	$O(1)$	$O(1)$	$O(n)$
Search / contains :	$O(1)$	$O(n)$	$O(\log_2 n)$
Delete / Remove :	$O(1)$	$O(n)$	$O(n)$

Real time Usage: Java Projects

1. Eliminating duplicates
2. Fast membership checks
3. Set operations: union, intersection and difference - find common elements, unique elements between sets
4. Implementing caches and lookup tables

The screenshot shows an IDE with a Java file named `Main.java`. The code demonstrates set operations using `HashSet`. A Venn diagram is drawn on the right side of the code editor, illustrating the intersection and difference of two sets, `set1` and `set2`.

Code Snippet:

```

67 HashSet<Integer> intersection = new HashSet<>(set1);
68 intersection.retainAll(set2);
69 System.out.println("Intersection: "+intersection);
70
71 // Difference: elements in set1 but not in set2
72 HashSet<Integer> difference = new HashSet<>(set1);
73 difference.removeAll(set2);
74 System.out.println("Difference: "+difference);
75 // Your task to find: set2-set1
76
77 // Symmetric Difference: elements in either set, but not in both
78 HashSet<Integer> symmetricDifference = new HashSet<>(set1);
79 symmetricDifference.addAll(set2);
80 HashSet<Integer> tempSet = new HashSet<>(set1);
81 tempSet.retainAll(set2);
82 symmetricDifference.removeAll(tempSet);
83 System.out.println("Symmetric Difference: "+symmetricDifference);
84
85 }

```

Venn Diagram:

The Venn diagram consists of two overlapping circles labeled `set1` and `set2`. The elements in the intersection of the two sets are 3 and 4. The elements in `set1` but not in `set2` are 1 and 2. The elements in `set2` but not in `set1` are 5 and 6.

Terminal Output:

```

Union: [1, 2, 3, 4, 5, 6]
Intersection: [3, 4]
Difference: [1, 2]
Symmetric Difference: [1, 2, 5, 6]

```

The terminal prompt shows the user is `ingledarshan@192` on `NS 20 June 2023`.