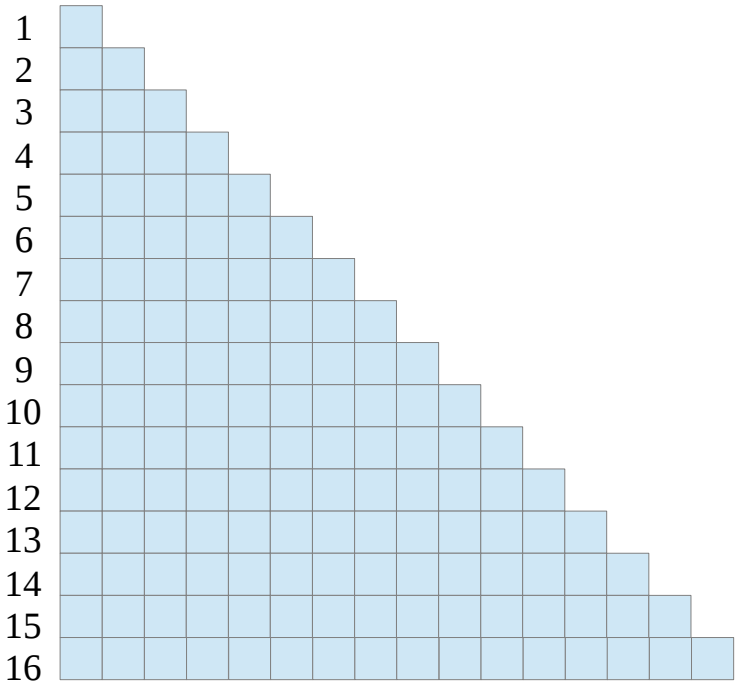
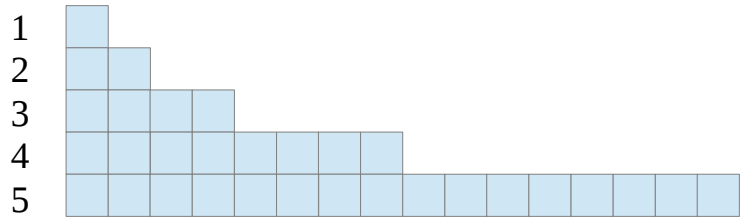
Algorithms – Binary Search (1D)

Please send corrections (or suggestions) to youngwlim@hotmail.com.

This document was produced by using LibreOffice and Octave.

O(**n**) vs. O(log **n**)





https://stackoverflow.com/questions/11032015/how-to-find-time-complexity-of-an-algorithm

Linear Search Algorithm

procedure linear search(x : integer, a_1, … , a_n : distinct integers)

i := 1

while (i ≤ **n** and x ≠ a_i)

    i := i + 1

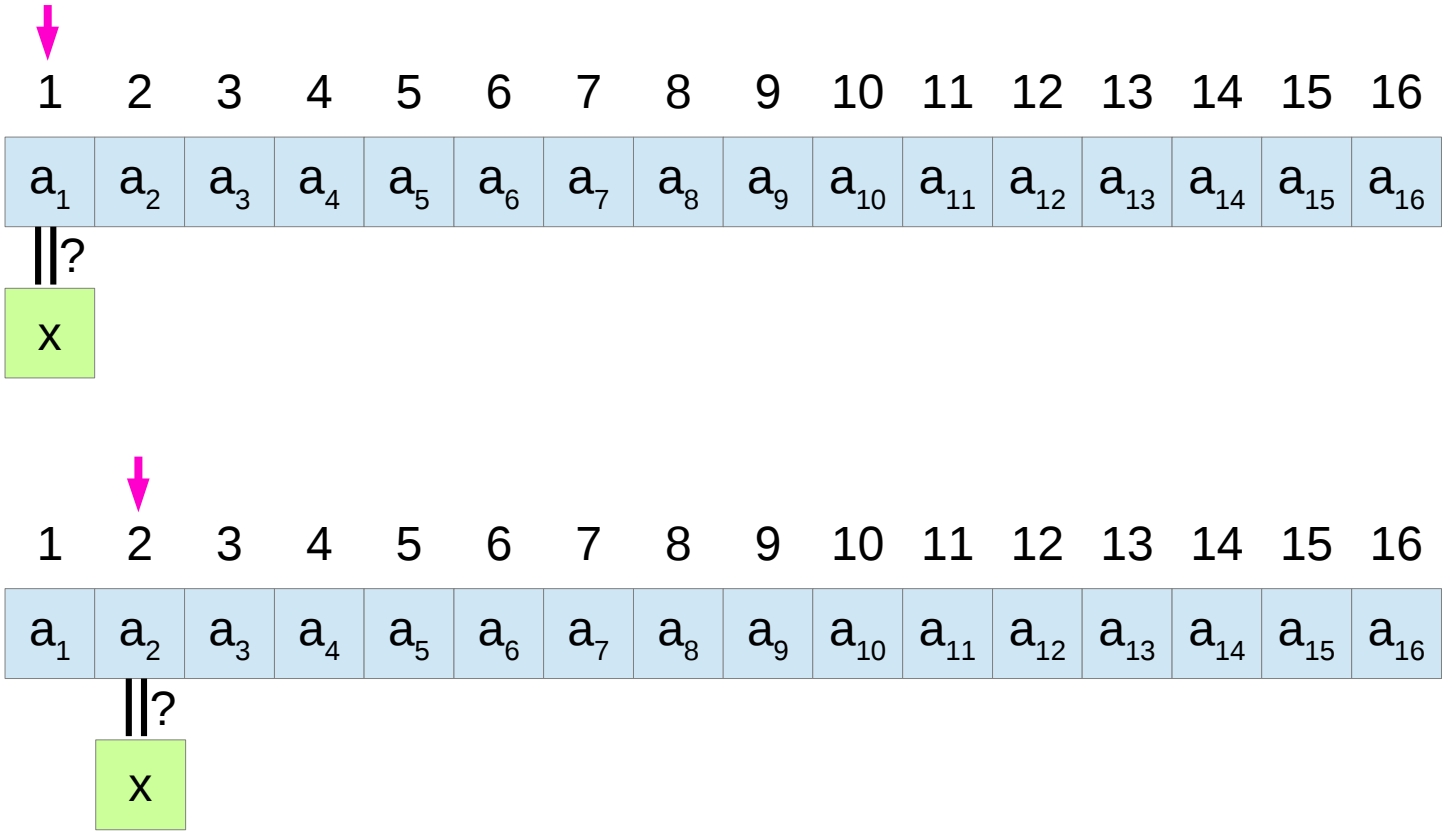
if (i ≤ **n**) **then** location := i

else location := 0

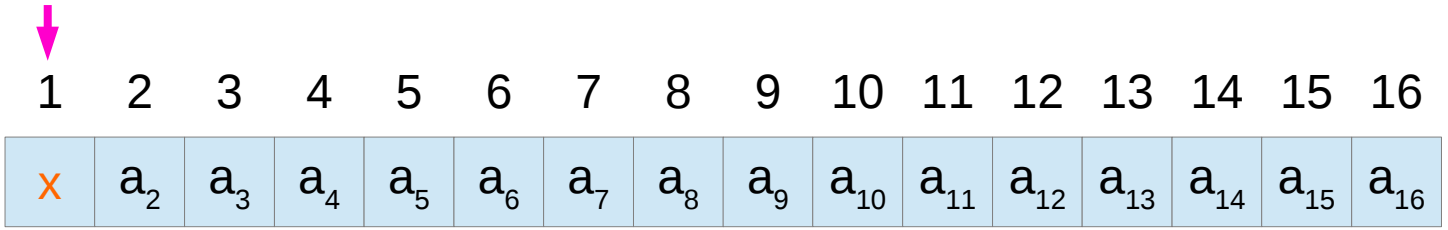
return location

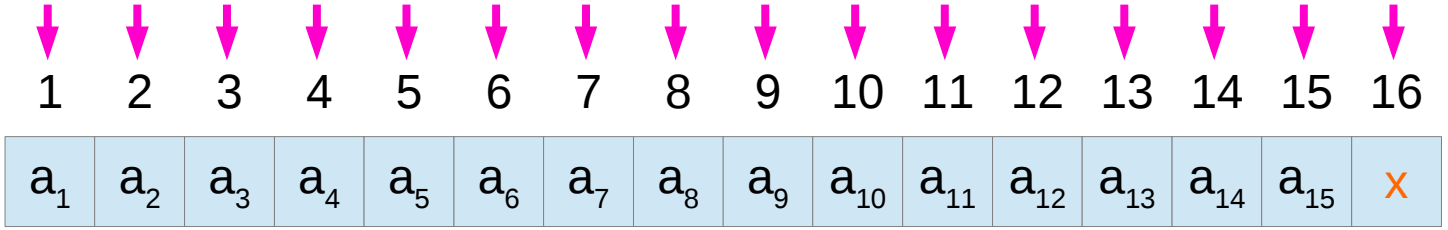
{location is the subscript of the term that equals x, or is 0 if x is not found}

i=1 and i=2


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12 a13 a14 a15 a16
?
x
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
a1 a2 a3 a4 a5 a6 a7 a8 a9 a10 a11 a12 a13 a14 a15 a16
?
x

Best and Worst Cases





Binary Search Algorithm

procedure binary search(x : integer, a_1, … , a_n : increasing integers)

i := 1 { i is left endpoint of search interval }

j := n { j is right endpoint of search interval }

while (i < j)

    m := **floor**((i+j)/2)

    if (x > a_m) **then** i := m + 1

    else j := m

if (x = a_i) **then** location := I

else location := 0

return location

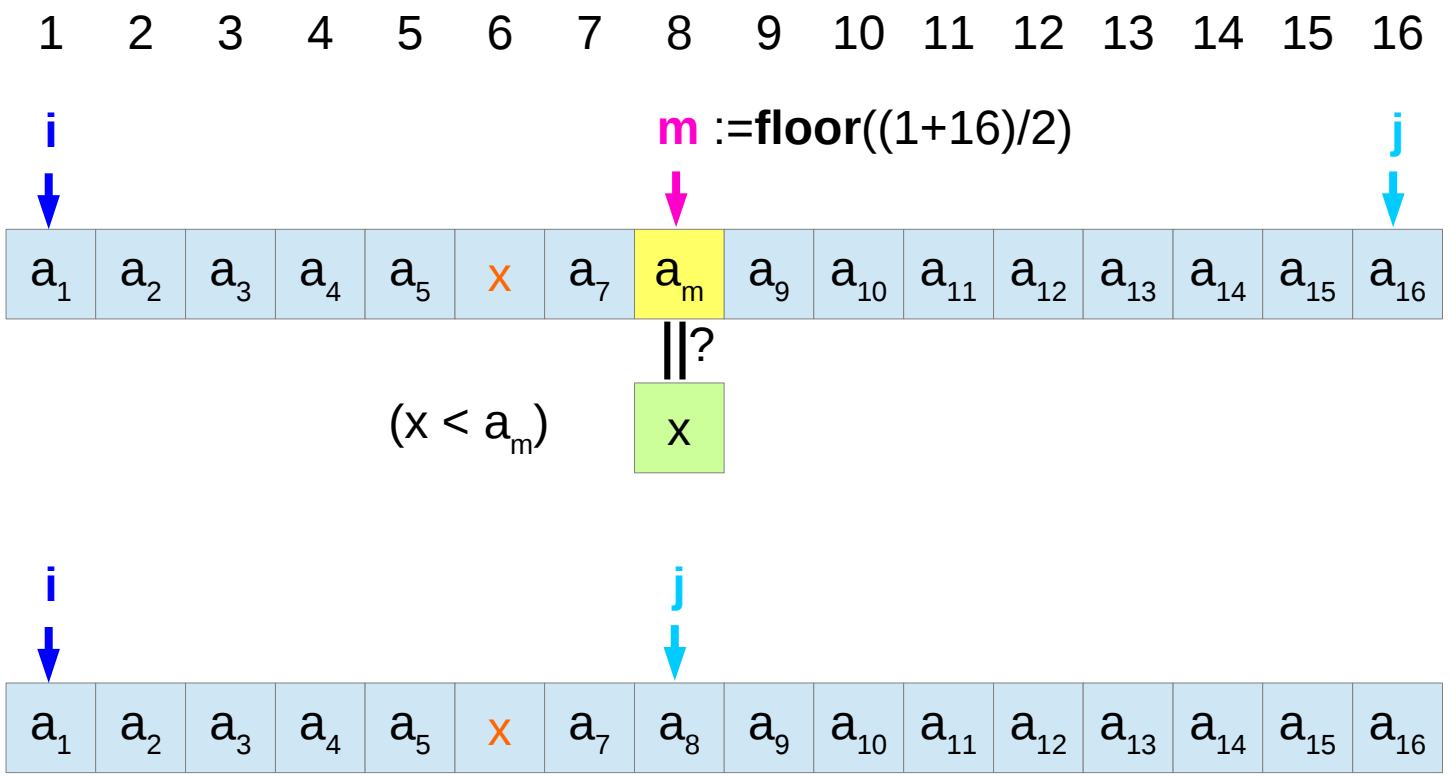
{location is the subscript of the term that equals x, or is 0 if x is not found}

Increasing Order Assumption

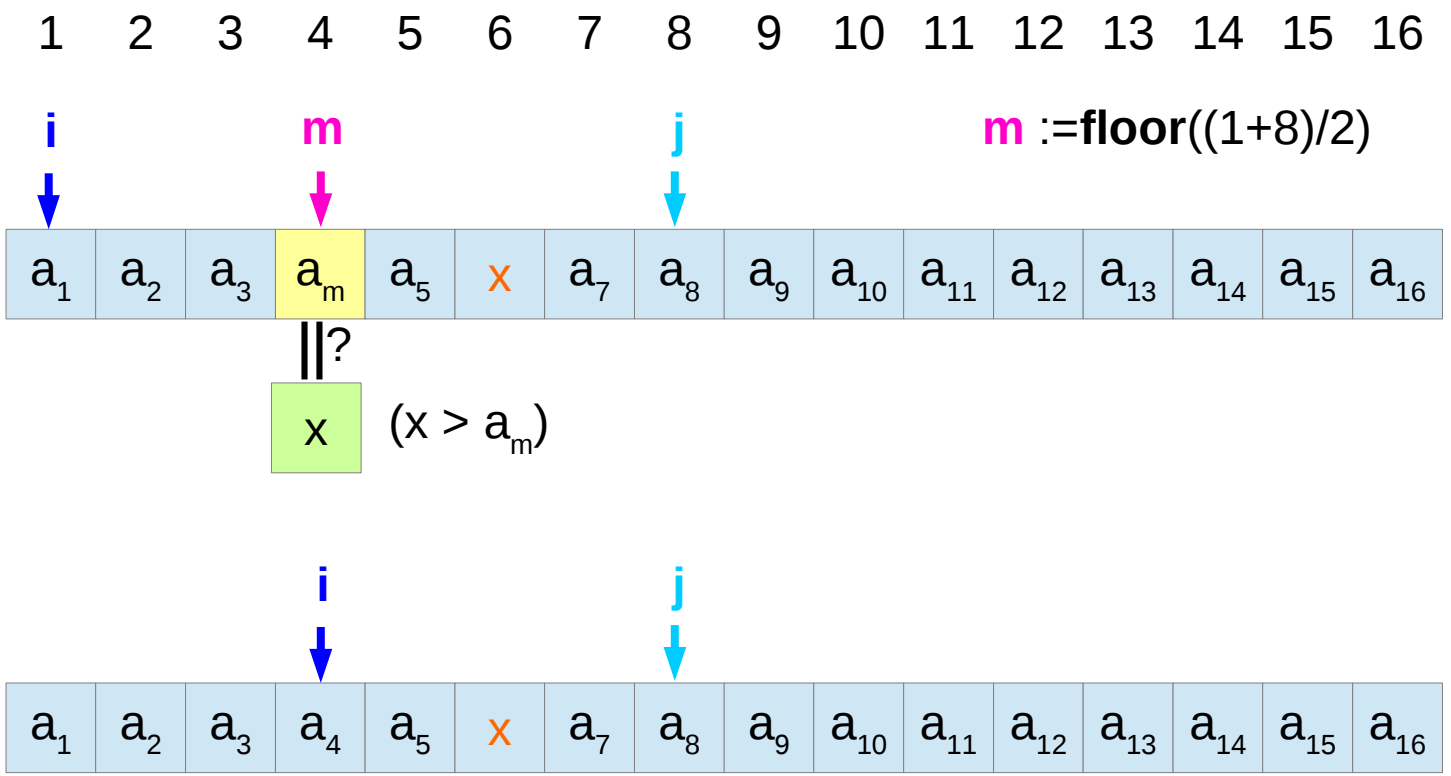
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}

$$a_1 \le a_2 \le a_3 \le a_4 \le a_5 \le a_6 \le a_7 \le a_8 \le a_9 \le a_{10} \le a_{11} \le a_{12} \le a_{13} \le a_{14} \le a_{15} \le a_{16}$$

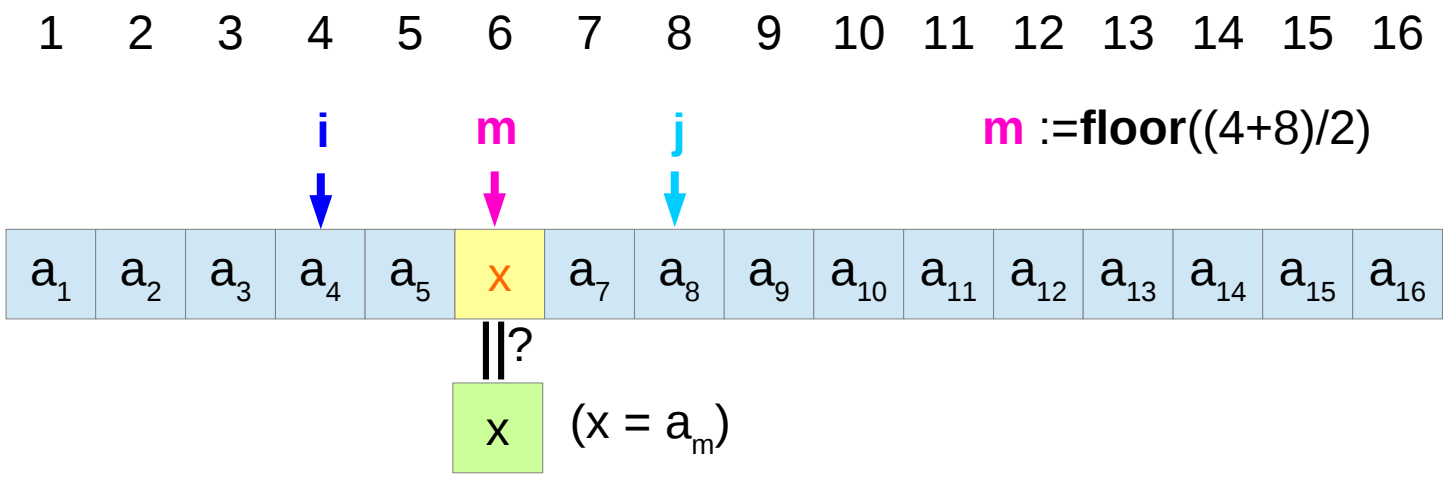
i=1


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
i
m :=floor((1+16)/2)
j
a1 a2 a3 a4 a5 x a7 am a9 a10 a11 a12 a13 a14 a15 a16
||?
(x < am)
x
i
j
a1 a2 a3 a4 a5 x a7 a8 a9 a10 a11 a12 a13 a14 a15 a16

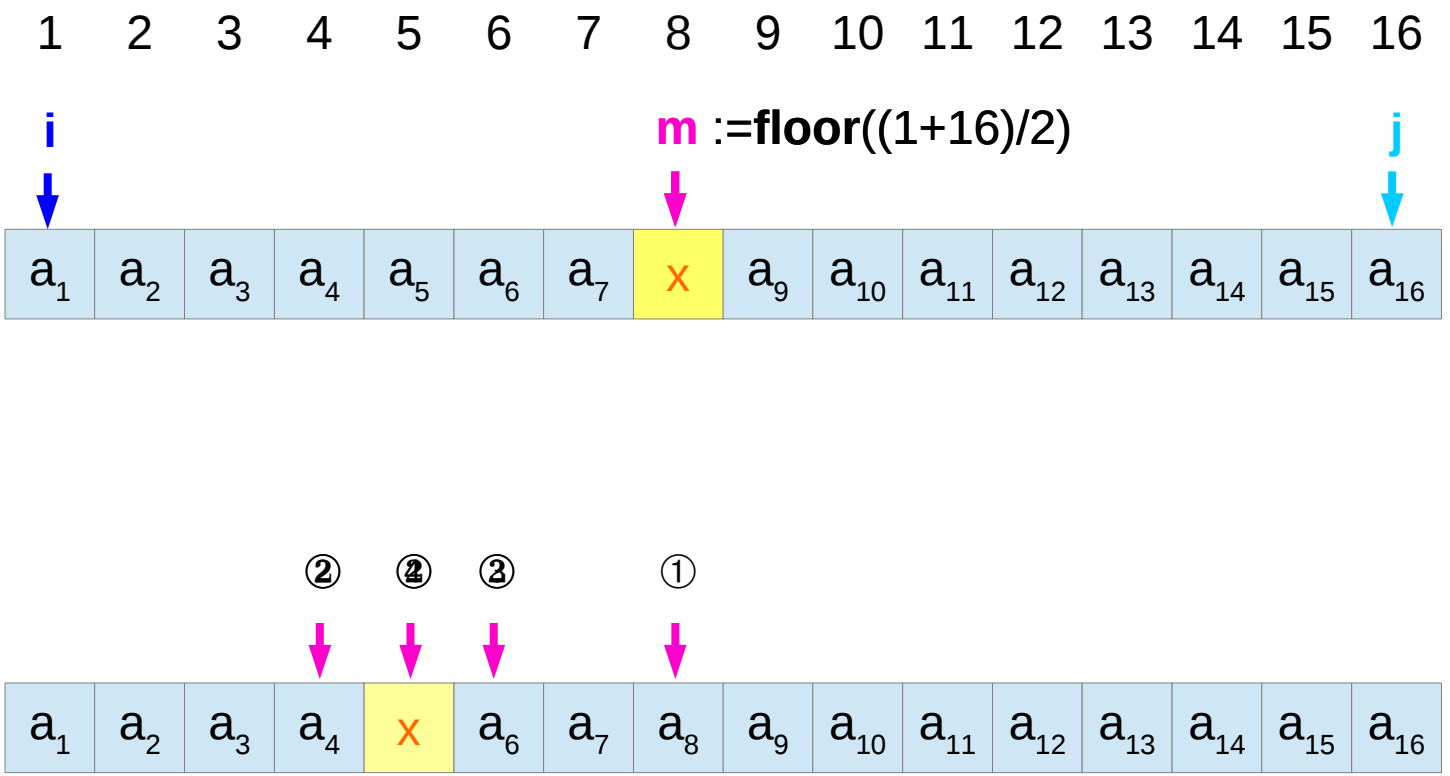
i=2


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
i
m
j
m :=floor((1+8)/2)
a1 a2 a3 am a5 x a7 a8 a9 a10 a11 a12 a13 a14 a15 a16
||?
x
(x > am)
i
j
a1 a2 a3 a4 a5 x a7 a8 a9 a10 a11 a12 a13 a14 a15 a16

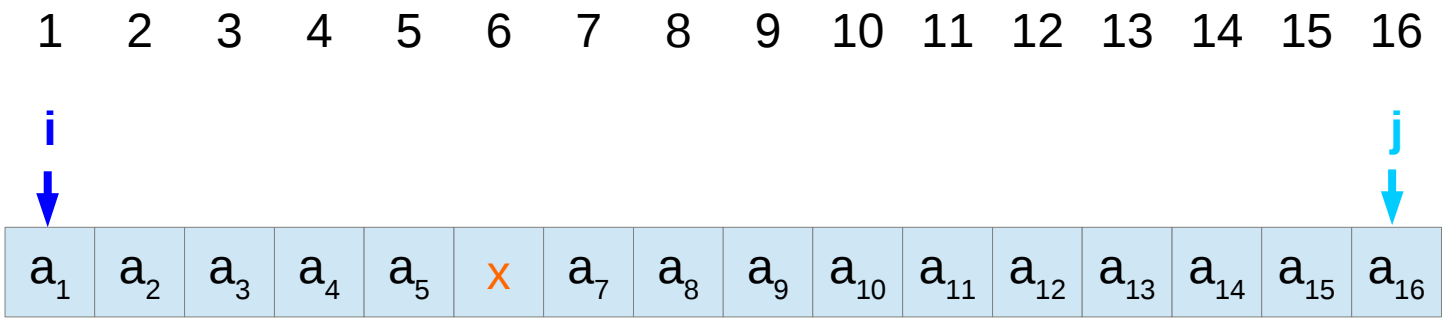
i=3


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
i
m
j
m :=floor((4+8)/2)
a1 a2 a3 a4 a5 x a7 a8 a9 a10 a11 a12 a13 a14 a15 a16
||?
x
(x = am)

Best and Worst Cases



Increasing Order



References

[1] http://en.wikipedia.org/
[2]