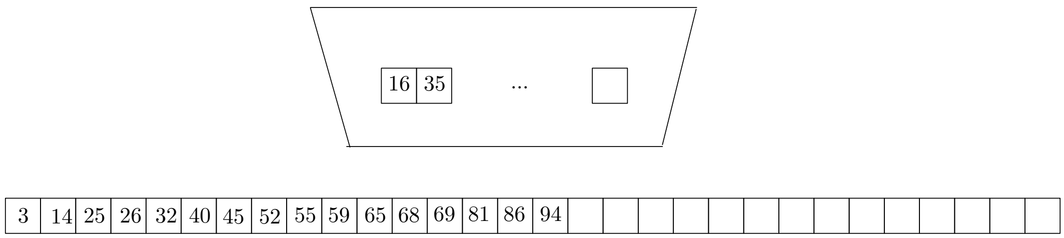
# CSCI2100: Regular Exercise Set 1

Prepared by Yufei Tao

**Problem 1.** Let $x$ be a real value. Define $\lfloor x \rfloor$ to be the largest integer that does not exceed $x$. For example, $\lfloor 2.5 \rfloor = 2$, whereas $\lfloor 3 \rfloor = 3$.

Suppose that you are given an integer $n \geq 2$ in (a register of) the CPU. Write an algorithm to compute the value of $\lfloor \log_2 n \rfloor$ in no more than $100 \log_2 n$ time.

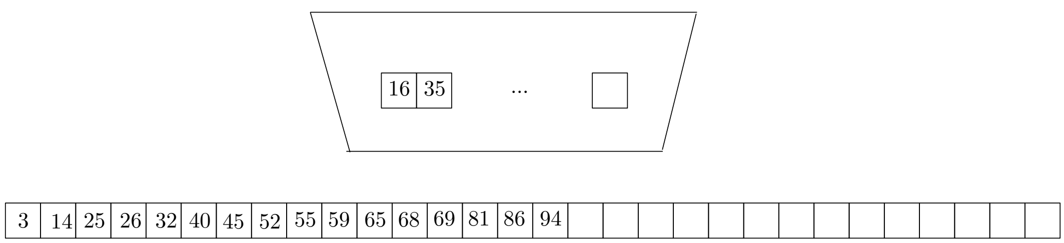
**Problem 2.** The following figure shows an input to the dictionary search problem.




Describe how binary search works using the input.

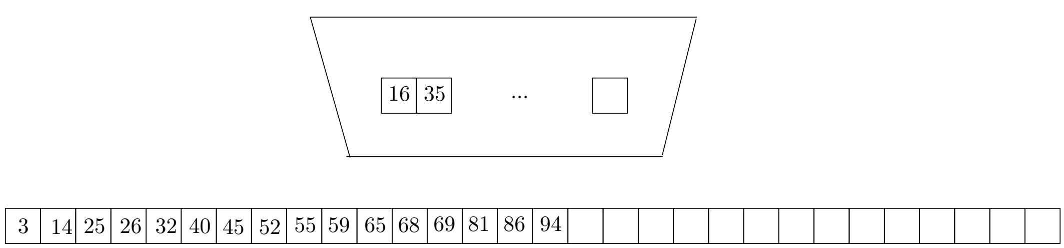
**Problem 3 (Predecessor Search).** Let us first define the notion of *predecessor*. Let $S$ be a set of integers. Given an integer $v$, the *predecessor* of $v$ in $S$ is the largest integer in $S$ that is at most $v$. For example, suppose $S = \{3, 14, 15, 26, 32, 40\}$. The predecessor of 25 is 15, while that of 26 is 26.

Consider the following problem. You are given a set $S$ of $n$ integers, which are stored at memory cells 1, 2, ..., $n$ in ascending order. The value of $n$ is given in the CPU, and so is an integer $v$. The following shows an example with $n = 16$ and $v = 35$.



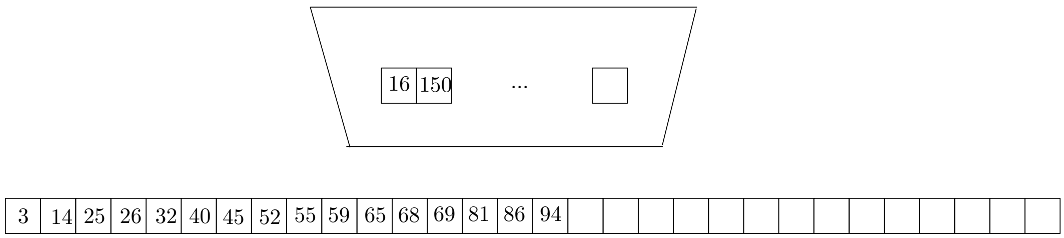

Describe an algorithm to find the predecessor of $v$. Your algorithm should have running time at most $100 + 100 \log_2 n$.

**Problem 4 (Prefix Counting).** Consider the following problem. You are given a set $S$ of $n$ integers, which are stored at memory cells 1, 2, ..., $n$ in ascending order. The value of $n$ is given in the CPU, and so is an integer $v$. The following shows an example with $n = 16$ and $v = 35$.



Describe an algorithm to find the number of integers in $S$ that are at most $v$. In the above example, for instance, you should return 5. Your algorithm should have running time at most $100 + 100 \log_2 n$.

**Problem 5 (The 3-Sum Problem).** Consider the following problem. The input $S$ consists of $n$ integers, which are given at memory cells 1, 2, ..., $n$, arranged in ascending order. The value of $n$ is given in the CPU. So is a value $v$. The following shows an example with $n = 16$ and $v = 150$.




Describe an algorithm to determine whether $S$ has 3 numbers that sum up to $v$. In the above example, the answer is "yes" because $150 = 40 + 45 + 65$. Your algorithm should have running time at most $100 + 100 \cdot n^2 \log_2 n$.