1. You are given a set \( A = \{a_1, a_2, \ldots, a_n\} \) of positive integers and a positive integer \( B \). A subset \( S \subseteq A \) is called feasible if the sum of the numbers in \( S \) does not exceed \( B \), i.e., \( \sum_{a_i \in S} a_i \leq B \). The sum of the numbers in \( S \) is called the total sum of \( S \).

You would like to select a feasible subset \( S \) of \( A \) whose total sum is as large as possible. For example, if \( A = \{8, 2, 4\} \) and \( B = 11 \) then the optimal solution is the subset \( S = \{8, 2\} \).

(a) Here is an algorithm for the problem.

\[
S \leftarrow \emptyset; \quad T \leftarrow 0
\]
for \( i \leftarrow 1 \) to \( n \) do
if \( T + a_i \leq B \) then
\[
S \leftarrow S \cup \{a_i\}
T \leftarrow T + a_i
\]
return \( S \)

Describe an input for which the total sum of the set \( S \) returned by this algorithm is less than half the total sum of some other feasible subset of \( A \).

(b) Describe a 1/2-approximation algorithm for the problem. Your algorithm should run in \( O(n \log n) \) time.

(c) Prove correctness of the algorithm you described in (b), i.e., it returns a feasible subset \( S \) whose total sum is at least half as large as the total sum of an optimal subset.

2. The Bin Packing problem takes as input an infinite supply of bins \( B_1, B_2, B_3, \ldots \), each bin of size 1 unit. We are also given \( n \) items \( a_1, a_2, \ldots, a_n \) and each item \( a_j \) has a size \( s_j \) that is a real number in the interval \([0,1]\). The Bin Packing problem seeks to find the smallest number of bins such that all \( n \) items can be packed into these bins.

For example, suppose that we are given 4 items \( a_1, a_2, a_3 \) and \( a_4 \) of sizes 0.5, 0.4, 0.6, and 0.5 respectively. We could pack \( a_1 \) and \( a_2 \) in bin \( B_1 \) because \( s_1 + s_2 = 0.9 \leq 1 \). We could then pack \( a_3 \) into bin \( B_2 \), but we could not also add \( a_4 \) to bin \( B_2 \), because \( s_3 + s_4 = 1.1 > 1 \). So \( a_4 \) would have to be packed in bin \( B_3 \). This gives us a bin packing of the 4 items into three bins. An alternate way of packing items that would lead to the use of just two bins
is to pack $a_1$ and $a_4$ into bin $B_1$ and $a_2$ and $a_3$ into bin $B_2$. This packing that uses only two bins is an optimal solution to the Bin Packing problem.

The *First Fit* greedy algorithm processes items in the given order $a_1, a_2, \ldots, a_n$ and it considers the bins in the order $B_1, B_2, \ldots$. For each item $a_j$ being processed, the algorithm packs $a_j$ into the first bin that has space for it.

Prove that the First Fit algorithm is a 2-approximation for Bin Packing.

3. Problem 1.1.
4. Problem 2.1.
5. Problem 3.1.
6. Problem 3.2.