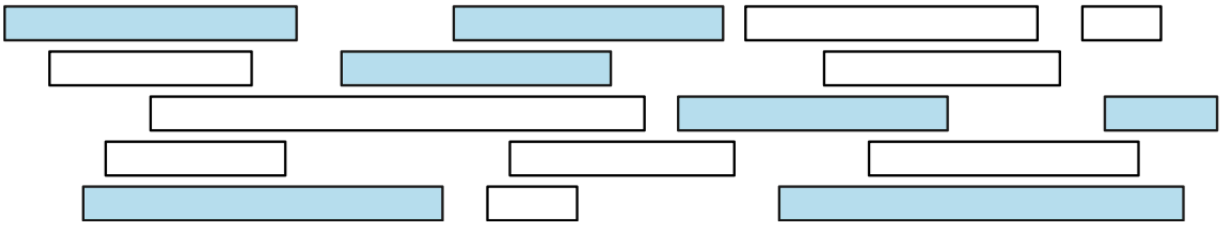
本次作业提交截止时间：2023年12月20日23:59

❗ 注意事项

- 每次作业请上传至对应的文件夹，比如第一次作业上传至文件夹homework1。
    - 该文件夹中包含两个文件夹：code和pdf。
    - 非代码请上传pdf文件至文件夹pdf中。非pdf（比如doc文件、md文件、tex文件）请转换成pdf。
    - C++代码请上传至文件夹code中。本课程只允许C++代码。
    - pdf文件和cpp文件都用自己的学号命名。比如1001.pdf和1001.cpp。
- 涉及公式的作业，推荐使用markdown编辑器。
- 涉及算法伪代码的作业，推荐使用在线$\LaTeX$编辑器Overleaf完成，提交编译生成的pdf文档即可。如果需要在overleaf中输入中文并能正确编译，首先在文档中加入package: `\usepackage[UTF8]{ctex}`，然后修改设置：设置->修改Latex引擎->选择“XeLatex”。
- 请基于提供的C++代码框架编写代码，注意：
    - 代码中不要包含任何中文
    - 不要使用<bits/stdc++.h>头文件
    - 不要在main函数后放置任何代码
    - 不要改变给定的函数原型
- 本次作业任选3题完成即可。

1. Let X be a set of n intervals on the real line. We say that a subset of intervals $Y \subseteq X$ *covers* X if the union of all intervals in Y is equal to the union of all intervals in X. The *size* of a cover is just the number of intervals.

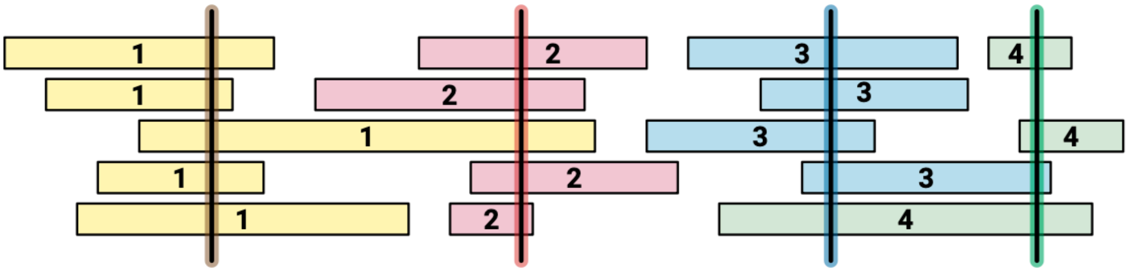
   Describe and analyze an efficient algorithm to compute the smallest cover of X. Assume that your input consists of two arrays $L[1..n]$ and $R[1..n]$, representing the left and right endpoints of the intervals in X. If you use a greedy algorithm, you must prove that it is correct.



A set of intervals, with a cover (shaded) of size 7.

2. Let X be a set of n intervals on the real line. We say that a set P of points *stabs* X if every interval in X contains at least one point in P.

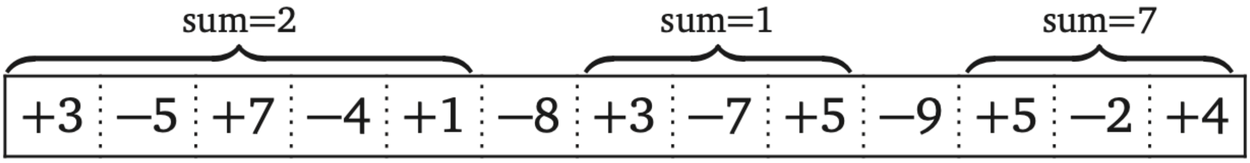
Describe and analyze an efficient algorithm to compute the smallest set of points that stabs X. Assume that your input consists of two arrays $L[1\,..\,n]$ and $R[1\,..\,n]$, representing the left and right endpoints of the intervals in X. As usual, if you use a greedy algorithm, you must prove that it is correct.



A set of intervals stabbed by four points (shown here as vertical segments)

3. Suppose you are given an array $A[1\,..\,n]$ of integers, each of which may be positive, negative, or zero. A contiguous subarray $A[i\,..\,j]$ is called a positive interval if the sum of its entries is greater than zero.

   Describe and analyze an algorithm to compute the minimum number of positive intervals that cover every positive entry in A. For example, given the following array as input, your algorithm should output 3. If every entry in the input array is negative, your algorithm should output 0. As usual, if you use a greedy algorithm, you must prove that it is correct.



4. A string w of parentheses (and) is **balanced** if it satisfies one of the following conditions:
   - w is the empty string.
   - $w = (x)$ for some balanced string x
   - $w = xy$ for some balanced strings x and y

   For example, the string $w = ((())()())(()())()$ is balanced, because $w = xy$, where $x = ((())()())$ and $y = (()())()$.
   (a) Describe and analyze an algorithm to determine whether a given string of parentheses is balanced.
   (b) Describe and analyze a greedy algorithm to compute the length of a longest balanced subsequence of a given string of parentheses. As usual, don't forget to prove your algorithm is correct.
   For both problems, your input is an array $w[1\,..\,n]$, where for each i, either $w[i] = ($ or $w[i] =)$. Both of your algorithms should run in $O(n)$ time.