Homework 04 (Due: Wednesday, April 23, 2025, 11 : 59 : 00PM (Eastern Time))

CPSC 3120

Instructions

This assignment consists of 4 analytical problems and 2 programming problems. Your solutions to the analytical problems must be submitted, as one PDF **without spaces, tabs, parentheses, pound signs, or percent signs in the filename**, via Canvas. While handwritten (then scanned) solutions to the analytical problems are acceptable, you are strongly encouraged to typeset your solutions in LaTeX or a word processor with an equation editor. The legibility of your solutions is of great importance.

Programming Assignment

Your methods will be tested on `newton.computing.clemson.edu`, using
`gcc version 9.4.0 (Ubuntu 9.4.0-1ubuntu1~20.04.1)` and be compiled for C++ 2017. To ensure proper execution, you should review the reports that will be sent back to you on Canvas.

You will submit `cpsc3120homework04part01.h`, `cpsc3120homework04part02.h`, `cpsc3120homework04part01` and `cpsc3120homework04part02.cpp`, along with your PDF, via Canvas.

`longestIncreasingSubsequence`

`longestIncreasingSubsequence` is a function that takes a `vector<int>` and returns an `int` (the length of the longest strictly-increasing subsequence in the `vector`). For example

```
::::::::::::::
part01test01.input
::::::::::::::
 -3  1 -7 -2 -5  3  0 -4 -8  2  7 -6 -9  4  9  6 -1  8  5
::::::::::::::
part01test01.solution
::::::::::::::
The longest increasing subsequence has a length of 7.
-3 -2 0 2 4 6 8
```

`minimumVotes`

`minimumVotes` is a function that takes a `vector<unsigned long long>` of popular votes cast, `vector<int>` of Electoral College votes allocated, and a `vector<string>` of state/district abbreviations and returns an `unsigned long long int` representing the minimum number of popular votes needed to win the given election.

The number of Electoral College votes assigned to each state/district for the 2028 United States Presidential Election will be based on the 2020 United States Census. Given a state's abbreviation, the number of Electoral College votes allocated to that state/district, and the (simulated) number of total votes cast in that state/district `minimumVotes` should return the fewest popular votes required for a candidate to win the (simulated) 2028 United States Presidential Election. You may assume that there are only two candidates and that winning a majority of the popular votes cast in a particular state/district is necessary and sufficient to win all of that state/district's Electoral College votes. You may also assume that a candidate is required to win a majority of 538 Electoral College votes available to win the election. For example

```
::::::::::::::
part02test01.input
::::::::::::::
AL       1384228   9
AK        440853   3
AZ       3677004  11
AR       1509561   6
CA       1723964  55
CO       2171870   9
CT        394724   7
DE        448232   3
DC         23078   3
FL       8604496  29
GA       1210262  16
HI        846085   4
ID        856728   4
IL       7536789  20
IN        660808  11
IA        140264   6
KS       1684376   6
KY       1169013   8
LA         68290   8
ME        355906   4
MD       1977882  10
MA       1406780  11
MI       4398235  16
MN       3490398  10
MS        853350   6
MO        903654  10
MT        442492   3
NE        771840   5
NV       1733339   6
NH        167999   4
NJ       1050152  14
NM        535743   5
NY       4647247  29
NC       2509090  15
ND        450061   3
```

```
OH       2649952  18
OK        795635   7
OR       2310916   7
PA       6545631  20
RI        484122   4
SC       3147316   9
SD        114251   3
TN       2698495  11
TX       1193007  38
UT       1328786   6
VT        325571   3
VA       4778444  13
WA       4918367  12
WV        505350   5
WI       3150367  10
WY         46750   3
::::::::::::::
part02test01.solution
::::::::::::::
Minimum required votes: 9972269
AL      9    692115
AK      3    220427
CA     55    861983
CT      7    197363
DE      3    224117
DC      3     11540
GA     16    605132
IN     11    330405
IA      6     70133
KY      8    584507
LA      8     34146
ME      4    177954
MA     11    703391
MS      6    426676
MO     10    451828
MT      3    221247
NH      4     84000
NJ     14    525077
NM      5    267872
ND      3    225031
OH     18   1324977
OK      7    397818
RI      4    242062
SD      3     57126
TX     38    596504
VT      3    162786
WV      5    252676
WY      3     23376
```

```
270    9972269
```

General Guidelines

Sample header, source, and testing files have been provided. You may modify the `.h` and `.cpp` files as needed, but you will only be turning in the four files mentioned above. The grading system will be compiling the code with the command
`g++ -std=c++17 -o /path/to/executable.out /path/to/source/files/*.cpp` for each part.

Written Assignment

Question 1 (10 points)
Question R-12.3 in *Algorithm Design and Applications*

Question 2 (10 points)
Question C-12.1 in *Algorithm Design and Applications*

Question 3 (10 points)
Question A-12.4 in *Algorithm Design and Applications*

Question 4 (10 points)
Question A-12.10 in *Algorithm Design and Applications*

Automated Report Notes

Reports will be generated every 3 minutes. Your programs should terminate within 60 seconds.

Point Allocation

Question	Points
Question 1	10%
Question 2	10%
Question 3	10%
Question 4	10%
`longestIncreasingSubsequence`	
Test Cases	1×20
Compilation	10
`longestIncreasingSubsequence` Total	30
`minimumVotes`	
Test Cases	1×20
Compilation	10
`minimumVotes` Total	30
Total	100%