University of Nevada, Las Vegas Computer Science 477/677 Spring 2023

Answers for Examination April 12, 2023

1. Fill in the blanks. [25 points]

   (a) The items in a priority queue represent **unfulfilled obligations**.

   (b) Name three kinds of search structures.

   Here are three that are commonly used. **list**, **hash table binary search tree**

2. [20 points] Write the prefix expression equivalent to the infix epression $-a * b - (-c - d) \wedge e$ (Don't forget that $\wedge$ means exponentiation.)

   $-* \sim ab \wedge - \sim cde$

   Some people wrote postfix instead. I gave partial credit. That answer is:

   $a \sim b * c \sim d - e \wedge -$

3. [20 points] Walk through the stack algorithm to change the infix expression $-a + b \wedge c \wedge -f$ to postfix. Show the stack at each step.

stack	infile	outfile
	$-a + b \wedge c \wedge -f$	
$\sim$	$a + b \wedge c \wedge -f$	
$\sim$	$+b \wedge c \wedge -f$	a
	$+b \wedge c \wedge -f$	$a \sim$
$+$	$b \wedge c \wedge -f$	$a \sim$
$+$	$\wedge c \wedge -f$	$a \sim b$
$+\wedge$	$c \wedge -f$	$a \sim b$
$+\wedge$	$\wedge - f$	$a \sim bc$
$+ \wedge \wedge$	$-f$	$a \sim bc$
$+ \wedge \wedge \sim$	f	$a \sim bc$
$+ \wedge \wedge \sim$		$a \sim bcf$
$+ \wedge \wedge$		$a \sim bcf \sim$
$+\wedge$		$a \sim bcf \sim \wedge$
$+$		$a \sim bcf \sim \wedge\wedge$
		$a \sim bcf \sim \wedge \wedge +$

4. [20 points] Up to now, no one has written a polynomial time algorithm for the subset sum problem, given below. However, there is a pseudopolynomial time algorithm. Write code or pseudocode for the pseudopolynomial time algorithm for deciding whether there is a subsequence of a given finite sequence of positive integers whose sum is a given integer.

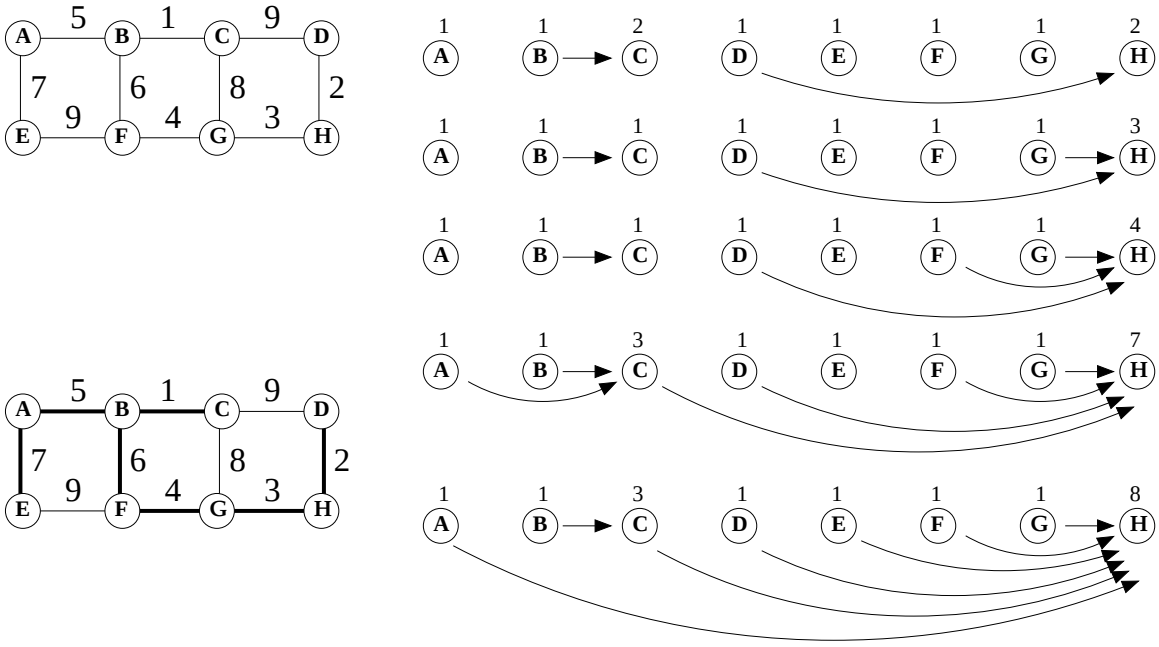
Let $x[1] \dots x[n]$ be the sequence, and K the given desired total.

```
bool S[n+1][K+1]; // S[i][j] means there is a subsequence of x[1]...x[i] whose total is j
S[0][0] = true; // the empty set is a solution if K = 0
for all j from 1 to K
   S[0][j] = false;
for all i from 1 to n // begin main outer loop
   for all j from 0 to K // begin main inner loop
     {
       if (S[i-1][j])
         {
           S[i][j] = true;
           if (x[i] + j ≤ K)
             [i][x[i] + j] = true;
         }
     }
if (S[n][K]) write "There is a solution.";
else write "There is no solution.";
```

The code below will write the solution, if $S[n][K]$ is true. Execute writesolution(n, K).

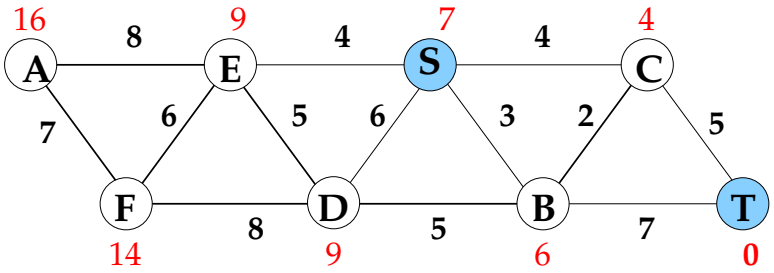
```
void writesolution(int i, int j)
// input condition: S[i][j]
   {
     if (i > 1)
       {
         if (S[i-1][j])
           writesolution(i-1, j);
         else
           {
             writesolution(i-1, j-x[i]);
             write x[i]; // x[i] is part of the subsequence
           }
       }
   }
```

5. [20 points] Walk through Kruskal's algorithm, using union/find, for the following weighted graph. Be sure to watch for path compression.

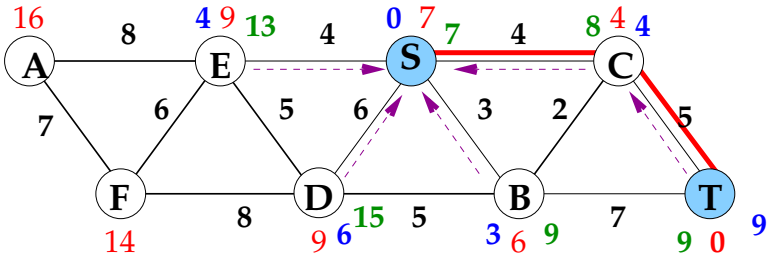


6. [20 points] Work the A^* algorithm for the following graph.

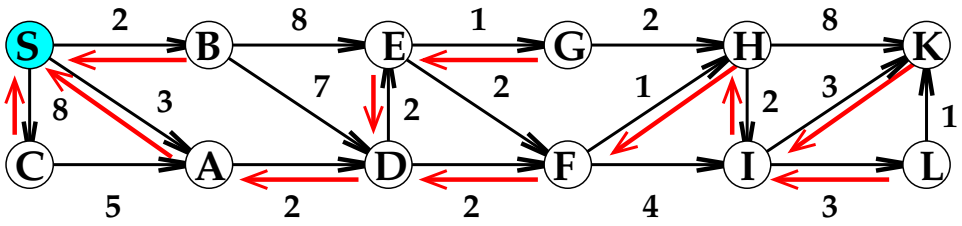
The heuristic function h is in red.



The function f is in blue, and the function g is in green.

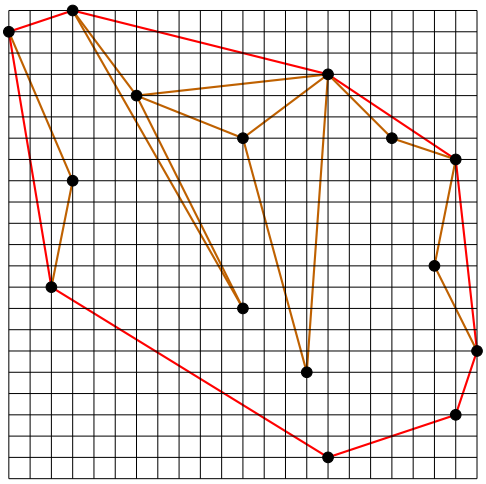


7. [20 points] Walk through Dijkstra's algorithm for the following graph.



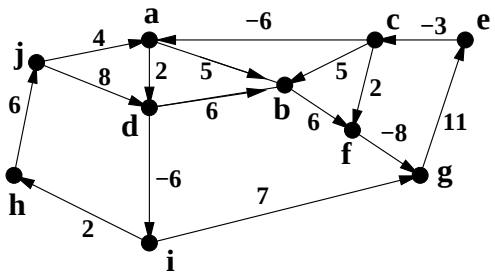
	S	A	B	C	D	E	F	G	H	I	K	L
V	0	3	2	8	~~9~~	5	~~10~~	7	7	8	8	~~11~~ 10	~~16~~ 13	13
back		S	S	S	~~B~~	A	~~B~~	D	D	E	F	~~F~~ H	~~H~~ I	I

8. [20 points] Walk through Graham Scan to find the convex hull of the points in the plane given in this figure.

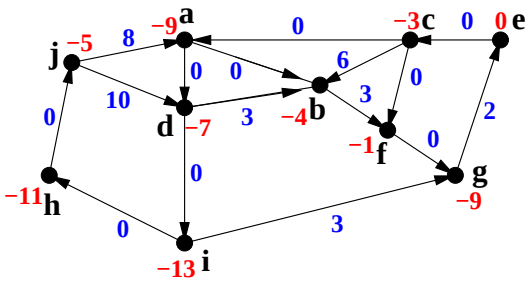


9. [20 points] Figure (a) below shows an instance of the all-pairs minpath problem. Work the first part of Johnson's algorithm on that graph, showing the adjusted weights in Figure (b).

Do not complete the computation of Johnson's algorithm.



(a)



(b)

10. [20 points] Walk through heapsort for the list BGHKRET.

1	2	3	4	5	6	7
B	G	H	K	R	E	T
B	G	T	K	R	E	H
B	R	T	K	G	E	H
T	R	B	K	G	E	H
T	R	H	K	G	E	B
B	R	H	K	G	E	T
R	B	H	K	G	E	T
R	K	H	B	G	E	T
E	K	H	B	G	R	T
K	E	H	B	G	R	T
K	G	H	B	E	R	T
E	G	H	B	K	R	T
H	G	E	B	K	R	T
B	G	E	H	K	R	T
G	B	E	H	K	R	T
E	B	G	H	K	R	T
B	E	G	H	K	R	T