ShanghaiTech University

CS101 Algorithms and Data Structures

Fall 2021

Homework 1

Due date: 23:59, September 26, 2021

1. Please write your solutions in English.
2. Submit your solutions to gradescope.com.
3. Set your FULL Name to your Chinese name and your STUDENT ID correctly in Account Settings.
4. If you want to submit a handwritten version, scan it clearly. Camscanner is recommended.
5. When submitting, match your solutions to the according problem numbers correctly.
6. No late submission will be accepted.
7. Violations to any of the above may result in zero grade.

1: Academic integrity (10 pts)

Question 1. *Please determine who violated academic integrity (or committed plagiarism) in the following situations. A stands for Alice, B stands for Bob, N stands for None, AB stands for Both.*

Note that you should write you answers of section 1 in the table below.

Q1 (a)	Q1 (b)	Q1 (c)	Q1 (d)	Q1 (e)
AB	N	AB	A	AB

(a) Alice and Bob are good friends, Alice asked if Bob could send her some code snippets so that she could have some ideas of what is going on in this homework (promising, of course, not copying his code). Bob agreed and sent his code repository to Alice, Alice copied the code and submitted to the CS 101 Platform.

(b) Bob is Alice's boyfriend. In order to enhance their romantic relationship, they studied together in the ShanghaiTech library and collaborated on an individual programming assignment. They discussed about the algorithm using the whiteboard but they didn't see any lines of code from each other.

(c) Alice and Bob are roommates. Bob let Alice use his laptop to play APEX Legends, and Alice copied Bob's solution for a paper assignment.

(d) Alice found the solution for a specific assignment on github and copied it to her homework with some trivial modifications.

(e) Alice borrows Bob's computer to complete her programming assignment. Without looking at Bob's code, she accidentally submits her own assignment using Bob's account to the CS 101 Platform. After that, Alice and Bob resubmitted their assignments separately using their own accounts.

2: (3*2') Multiple Choices

Each question has one or more correct answer(s). Select all the correct answer(s). For each question, you get 0 point if you select one or more wrong answers, but you get 1 point if you select a non-empty subset of the correct answers.

Note that you should write you answers of section 2 in the table below.

Question 2	Question 3	Question 4
A	D	A

Question 2. *Using a single linked list to implement a stack, suppose you only keep the head of it, where should the pushes and pops be performed in the optimal way?*

(A) Push in front of the first element, pop the first element

(B) Push in front of the first element, pop the last element

(C) Push after the first element, pop the first element

(D) Push after the last element, pop the last element

(E) Push after the last element, pop the first element

Question 3. *Suppose we use a circular array with index range from 0 to N - 1 to implement a queue, and currently Front is pointing at index m. We will know that this queue is full if* ***Back*** *is pointing at index =* _____. *(Options below are* ***Integers Modulo N****:* $\mathbb{Z}_N = \{0, 1, 2, ..., N-1\}$*)*

(A) 0

(B) m

(C) $N-1$

(D) $m-1$

Question 4. *What function does the following code achieve?*

```
void Q4(Queue &Q)
{
  Stack S;
  int d;
  S.InitStack();
  while(!Q.IsEmpty())
  {
    d = Q.DeQueue();
    S.Push(d);
  }
  while(!S.IsEmpty())
  {
    d = S.Pop();
```

```
      Q.EnQueue(d);
    }
  }
```

(A) Use a stack to reverse a queue.

(B) Use a queue to reverse a stack.

(C) Use a stack to implement a queue.

(D) Use a queue to implement a stack.

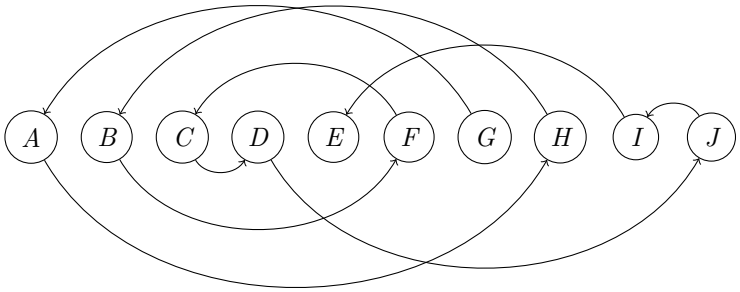
3: Array for list (10 pts)

Question 5. *In this question, we use two arrays: "next" and "value" to simulate a linked-list. The elements at each index in "next" and "value" are combined together to represent a node in the linked-list, where "next" represents the pointer to the next node, and "value" represents data in the nodes.*

(a) ***Linked list from array representation (5 pts)***

Here are the two arrays of "next" and "value". Please draw the linked list presented by the two arrays.

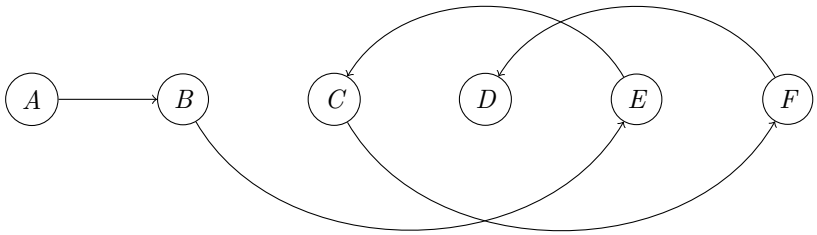
index	0	1	2	3	4	5	6	7	8	9
value	A	B	C	D	E	F	G	H	I	J
next	7	5	3	9	/	2	0	1	4	8



$$G \longrightarrow A \longrightarrow H \longrightarrow B \longrightarrow F \longrightarrow C \longrightarrow D \longrightarrow J \longrightarrow I \longrightarrow E$$

(b) ***Array representation for a linked list (5 pts)***

Fill in the "next" array in order to make the array and the linked-list showing below equivalent.



index	0	1	2	3	4	5
value	A	B	C	D	E	F
next	1	4	5	/	2	3

4: (8') Postfix expression

Question 6. *Reverse Polish notation (RPN) is a mathematical notation in which operators follow their operands. Using a stack, we can evaluate postfix notation equations easily.*

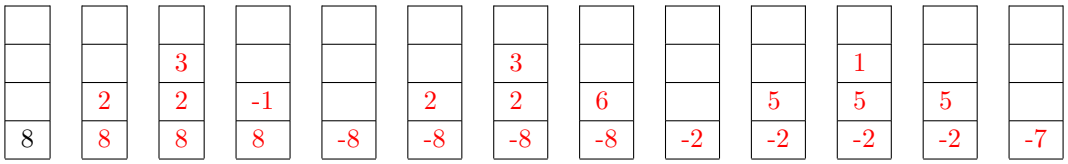
(a) ***Calculating(1'+2')***

A post-fix expression (Reverse-Polish Notation) with single digit operands is shown below: (where ^ is the exponentiation operator)

8 2 3 ^ / 2 3 * + 5 1 * −

Its in-fix expression is: $8/2\hat{}3+2*3-5*1$

The changing of the stack to calculate the final result is:



(b) ***Conversion(2*0.5'+2*1')*** *Now, try to convert the following in-fix expression into post-fix expression: (You don't need to calculate them)*

1) $1+2+3$
1 2 + 3 +

2) $1-2/3$
1 2 3 / −

3) $1+2*3+(4*5+6)*7$
1 2 3 * + 4 5 * 6 + 7 * +

4) $1+(2*3\hat{}4)/(5+6)+7$
1 2 3 4 ^ * 5 6 + / + 7 +

(c) ***Validity(4*0.5')***

Please judge whether the following post-fix expression is legal, if legal, please write ***T****, otherwise please write* ***F****.*

(a) 1 2 * + 3 5 + *F*

(b) 4 5 6 / * 1 / *T*

(c) 1 + 2 − 3 + 4 *F*

(d) 7 8 9 1 + − * *T*

5: (5'+5') Big-O Notation

Question 7. ***(5') Asymptotic Analysis***

For each pair of functions $f(n)$ and $g(n)$, give your answer whether $f(n) = O(g(n))$, $f(n) = \Omega(g(n))$ or $f(n) = \Theta(g(n))$. Give a ***proof*** *of your answers.*

Note 1: Try to give your answer in the most precise form. For example, for $f(n) = n^2$ and $g(n) = n^2+2n+1$, write $f(n) = \Theta(g(n))$.

Note 2: Try to prove by calculating limits.

1. $f(n) = 1.01^n$ *and* $g(n) = n^{10}$

   $f(n) = \Omega(g(n))$

   Note: Exponential dominates Polynomial.

   Using L'Hopital's rule.

$$\lim_{x\to\infty} \frac{f(x)}{g(x)} = \lim_{x\to\infty} \frac{1.01^x}{x^{10}} = \lim_{x\to\infty} \frac{\ln 1.01 \cdot 1.01^x}{10x^9} = ... = \lim_{x\to\infty} \frac{1.01^x}{constant} = \infty$$

2. $f(n) = \log(n!)$ *and* $g(n) = \log(n^n)$

   $f(n) = \Theta(g(n))$

   Note: You can also refer to [Stirling formula] for details.

   Another way to prove: (by-definition)

$$g(n) = \log(n^n) = n\log n = \Theta(n\log n)$$

$$f(n) = \log(n!) = \sum_{i=1}^{n} \log(i) \geq \sum_{i=\frac{n}{2}}^{n} \log(i) \geq \sum_{i=\frac{n}{2}}^{n} \log(\frac{n}{2}) = (\frac{n}{2})\log(\frac{n}{2})$$

   and

$$f(n) = \log(n!) = \sum_{i=1}^{n} \log(i) \leq n\log n$$

   hence,

$$f(n) = \Theta(n\log n) = \Theta(g(n))$$

3. $f(n) = \log^2(n)$ *and* $g(n) = \log\log n$

   $f(n) = \Omega(g(n))$

   Note: $\log^2 n = \Omega(\log n), \log n = \Omega(\log\log n)$. *You can prove it by calculating the limits.*

Question 8. ***(5') Complexity Analysis***

Consider the factorial computing problem: $N! = 1 \times 2 \times \cdots \times N$.

1. *(2') To store the number* N*, we need at least* $\log N$ *bits of space. Find an* $f(N)$ *such that* $N!$ *is* $\Theta(f(N))$ *bits long. Simplify your answer and justify your answer.*

   Solution:
   Method 1:
   Using Stirling's formula, you may get $N \log N$ *bits.*

   Method 2:
   When we multiply an m *bit number by an* n *bit number, we get an* $(m+n)$ *bit number. When computing factorials, we multiply* N *numbers that are at most* $\log N$ *bits long, so the final number has at most* $N \log N$ *bits. But if you consider the numbers from* $\frac{N}{2}$ *to* N*, we multiply at least* $\frac{N}{2}$ *numbers that are at least* $\log N - 1$ *bits long, so the resulting number has at least* $\frac{N(\log N - 1)}{2}$ *bits.*

2. *(3') Consider this naive factorial computing algorithm.*

   procedure *FACTORIAL(*N*)*
   if $N > 1$ ***then***
   return $FACTORIAL(N-1) \cdot N$
   else
   return 1
   end procedure

   We assume the runtime of multiplying an m*-bit number and an* n*-bit number is* $\Theta(mn)$*. What's the runtime of this factorial computing algorithm in Big-O notation? Justify your answers.*

   Solution: *Running time :* $(N^2 \log^2 N)$.
   We have N *recursive calls, each you can think of multiplying an* $N \cdot \log N$*-bit number (at most) by an* $\log N$*-bit number. Using the naive multiplication algorithm, each multiplication takes time* $\mathcal{O}(N \cdot log^2 N)$*. Hence, the running time is* $\mathcal{O}(N^2 \log^2 N)$.

   A lower bound of $\Omega(N^2 \cdot \log^2 N)$ *follows because the product of the first* $N/2$ *numbers will be at least* $\Omega(N \log N)$ *bits long by the previous part, and the last* $N/2$ *multiplications take* $\Omega(N \log^2 N)$ *time each.*