CSCI2100D 2023-24: Solution 1[*]

[#] This assignment is due at 11:59:59pm, 16th February 2024.

- **Q1. [10 marks]** Consider the algorithm TWOSUM shown below. What is the worst case time complexity (using **Big-Oh**) of this algorithm? Justify your answer.

```
Algorithm twoSum(arr, n, target)
1:  numPairs = 0
2:  for i = 0 to n − 2
3:      for j = i + 1 to n − 1
4:          if arr[i] + arr[j] == target
5:              numPairs = numPairs + 1
6:  return numPairs
```

# of basic operations	
1:	$O(1)$
2:	$O(n)$
3:	$O(n^2)$
4:	$O(n^2)$
5:	$O(n^2)$
6:	$O(1)$

Therefore, the total time complexity of SELECTIONSORT is $O(max(1, n, n^2, n^2, n^2, 1))$=$O(n^2)$, according to the sum property.

- **Q2. [24 marks]** Answer the following questions related to $O(\cdot)$, $\Omega(\cdot)$ and $\Theta(\cdot)$.
  - **(i). [5 marks]** What is the asymptotic (**Big-Oh**) complexity of the function $g(n) = (n^3 + n^2 \log n) \cdot (n^2 + n\sqrt{n})$?

    Let $g_1(n) = n^3 + n^2 \log n$, $g_2(n) = n^2 + n\sqrt{n}$. Then $g_1(n) = O(n^3)$, $g_2(n) = O(n^2)$. According to the product rule, $g(n) = g_1(n) \cdot g_2(n) = O(n^3 \cdot n^2) = O(n^5)$.
  - **(ii). [5 marks]** What is the asymptotic (**Big-Theta**) complexity of the function $g(n) = (n^4 + 12n^2 + 5n) \cdot (3n + 6)$?

    Let $g_1(n) = n^4 + 12n^2 + 5n$, $g_2(n) = 3n + 6$. Then $g_1(n) = O(n^4), g_2(n) = O(n)$ and $g_1(n) = \Omega(n^4), g_2(n) = \Omega(n)$. According to the product rule, we have $g(n) =$

[*]Departmental Guideline for Plagiarism (Department of Systems Engineering and Engineering Management): If a student is found plagiarizing, his/her case will be reported to the Department Examination Panel. If the case is proven after deliberation, the student will automatically fail the course in which he/she committed plagiarism. The definition of plagiarism includes copying of the whole or parts of written assignments, programming exercises, reports, quiz papers, mid-term examinations and final examinations. The penalty will apply to both the one who copies the work and the one whose work is being copied, unless the latter can prove his/her work has been copied unwittingly. Furthermore, inclusion of others' works or results without citation in assignments and reports is also regarded as plagiarism with similar penalty to the offender. A student caught plagiarizing during tests or examinations will be reported to the Faculty office and appropriate disciplinary authorities for further action, in addition to failing the course.

$g_1(n) \cdot g_2(n) = O(n^4 \cdot n) = O(n^5)$ and $g(n) = g_1(n) \cdot g_2(n) = \Omega(n^4 \cdot n) = \Omega(n^5)$. Hence we can derive $g(n) = \Theta(n^5)$.

- **(iii). [7 marks]** Using the basic definitions, prove that $\sum_{i=1}^{n} i^n = O(n^{n+1})$.

  We have that: $\sum_{i=1}^{n} i^n \leq \sum_{i=1}^{n} n^n = n \times n^n = n^{n+1}$. So $\sum_{i=1}^{n} i^n = O(n^{n+1})$.
- **(iv). [7 marks]** Let $f(n)$ be a non-negative function. Using the basic definitions, prove that $max(f(n), O(f(n))) = \Theta(f(n))$.

  Consider $h(n) = max(f(n), g(n))$ where $g(n) = O(f(n))$. We can easily find that $h(n) \geq f(n)$, which means $h(n) = \Omega(f(n))$. Assume that c is a constant such that $0 \leq g(n) \leq c \cdot f(n)$ for all $n \geq n_0$. Then, Let $c' = max(1, c)$. We have that $h(n) \leq c' \cdot f(n)$ for all $n \geq n_0$, which means $h(n) = O(f(n))$. So $h(n) = max(f(n), O(f(n))) = \Theta(f(n))$.

- **Q3. [15 marks]** Use the master method to give asymptotic (**Big-Oh**) bounds for the following recurrences.
  - **(i). [5 marks]** $g(1) = c_0$, $g(n) \leq 3g(\lceil n/2 \rceil) + 2n^2$.

    We have that $a = 3$, $b = 2$, $\lambda = 2$. Since $\log_b a < \lambda$, $g(n) = O(n^\lambda) = O(n^2)$.
  - **(ii). [5 marks]** $g(1) = c_0$, $g(n) \leq 8g(\lceil n/2 \rceil) + 2n^3$.

    We have that $a = 8$, $b = 2$, $\lambda = 3$. Since $\log_b a = \lambda$, $g(n) = O(n^\lambda \cdot \log n) = O(n^3 \cdot \log n)$.
  - **(iii). [5 marks]** $g(1) = c_0$, $g(n) \leq 2g(\lceil n/4 \rceil) + 4$.

    We have that $a = 2$, $b = 4$, $\lambda = 0$. Since $\log_b a > \lambda$, $g(n) = O(n^{\log_b a}) = O(n^{\log_4 2}) = O(\sqrt{n})$.

- **Q4. [10 marks]** For the following functions, sort them in **nonincreasing** order of the growth rate. (Hint. If $f_1 = O(f_2)$, f_1 grows no faster than f_2.)

$$f_1(n) = n^{1.4} \log n, \quad f_2(n) = (\log n)^{10}, \quad f_3(n) = 1.0001^n, \quad f_4(n) = n^{1.5}.$$

Recall that the logarithm function is beaten by the polynomial function. We have $f_2(n) = (\log n)^9 \times \log n = O(n^{1.4} \times \log n) = O(f_1(n))$, and $f_1(n) = n^{1.4} \log n = O(n^{1.4} \cdot n^{0.1}) = O(f_4(n))$. Since the exponential function $f_3(n)$ is much faster than the polynomial functions, the correct order of these functions is $f_3(n), f_4(n), f_1(n), f_2(n)$.

- **Q5. [16 marks]** Given that $T(1) = 1$, answer the following questions.
  - **(i). [8 marks]** Show that the solution of $T(n) = T(n-1) + 2n$ is $O(n^2)$.

    We have that: $T(n) = T(n-1)+2n = T(n-2)+2n+2(n-1) = \cdots = T(1)+2n+2(n-1)+\cdots+2\times 2 \leq 2n\cdot(n-1)+1$. When $n \geq 1$, we can find $T(n) \leq 2n\cdot(n-1)+1 \leq 3\times n^2$. So $T(n) = O(n^2)$.
  - **(ii). [8 marks]** Show that the solution of $T(n) = n \cdot T(n-1)$ is $O(n^n)$.

    We have that: $T(n) = T(n-1) \times n = T(n-2) \times n \times (n-1) = \cdots = n \times (n-1) \times \cdots \times 2 \times T(1) \leq n \times n \times \cdots \times n = n^n$. So $T(n) = O(n^n)$

- **Q6. [15 marks]** Answer the following questions about the linked list.
  - **(i). [5 marks]** Assume that we have a linked list: *head* $\leftrightarrow 1 \leftrightarrow 7 \leftrightarrow 4 \leftrightarrow 9 \leftrightarrow$ *tail* as shown in Fig. 1. Fill the values in the question marks.

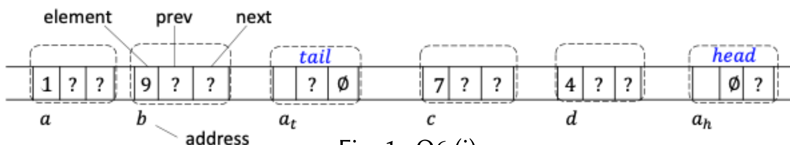

Fig. 1. Q6 (i)

Solution:

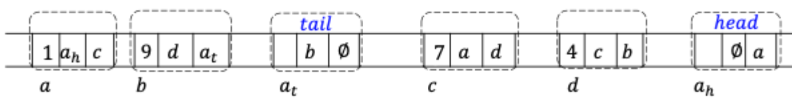

Fig. 2. Solution of Q6 (i)

  - **(ii). [5 marks]** We have a linked list: *head* $\leftrightarrow 6 \leftrightarrow 2 \leftrightarrow 4 \leftrightarrow$ *tail* as shown in Fig. 3. Assume that we insert element 5 to the tail of the list, that is, after 4, and the address of the node is a. Update the values after the insertion.

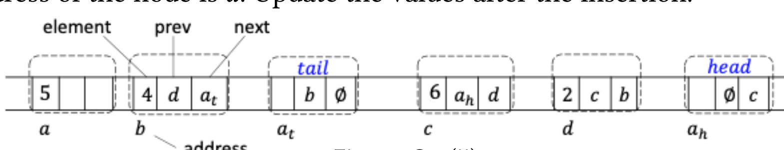

Fig. 3. Q6 (ii)

Solution:

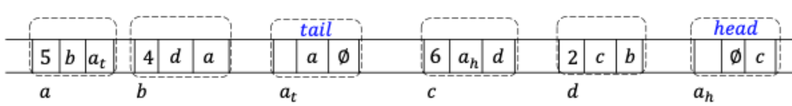

Fig. 4. Solution of Q6 (ii)

  - **(iii). [5 marks]** We have a linked list: *head* $\leftrightarrow 4 \leftrightarrow 3 \leftrightarrow 6 \leftrightarrow 9 \leftrightarrow$ *tail* as shown in Fig. 5. Update the values after deleting the node storing element 9 assuming we know its address b.

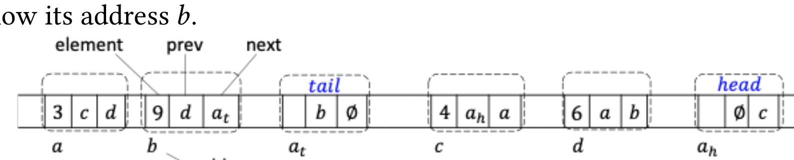

Fig. 5. Q6 (iii)

Solution:

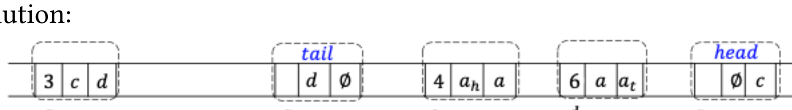

Fig. 6. Solution of Q6 (iii)

- **Q7. [10 marks]** Answer the following questions about stacks. Initially, the stacks are empty.

- **(i). [3 marks]** Assume the sequence of pushed-in elements is $[1, 3, 2, 4, 5, 6]$, and the sequence of stack operations is [PUSH, PUSH, POP, PUSH, PUSH, POP, POP, PUSH, PUSH, POP, POP, POP]. What is the sequence of popped-out elements?
  - **Example.** Consider the sequence of pushed-in elements $[3, 1, 2]$, and the sequence of stack operations [PUSH, PUSH, POP, PUSH, POP, POP]. That means the stack operations are [PUSH(3), PUSH(1), POP( ), PUSH(2), POP( ), POP( )]. Then the relative sequence of popped-out elements, which consists of the popped elements of each POP operation in order, will be $[1, 2, 3]$.

  Solution: $[3, 4, 2, 6, 5, 1]$.
- **(ii). [2 marks]** Assuming that the size of the stack is not infinite, an error will occur if the number of elements in the stack exceeds the size of the stack. In (i), what is the minimum possible stack size required to ensure that all operations are error-free? This means, that during the process, what is the maximum number of elements in the stack?
  - **Example.** Consider the sequence of pushed-in elements $[3, 1, 2]$, and the sequence of stack operations [PUSH, PUSH, POP, PUSH, POP, POP]. That means:
    1. After PUSH(3), the elements in the stack from bottom to top are $[3]$.
    2. After PUSH(1), the elements in the stack from bottom to top are $[3, 1]$.
    3. After POP( ), the elements in the stack from bottom to top are $[3]$.
    4. After PUSH(2), the elements in the stack from bottom to top are $[3, 2]$.
    5. After POP( ), the elements in the stack from bottom to top are $[3]$.
    6. After POP( ), the elements in the stack from bottom to top are $[]$.

    During the entire process, the minimum possible number of elements in the stack is 2, so the size of the stack is at least 2.

  Solution: 3.
- **(iii). [3 marks]** Assume the sequence of pushed-in elements is $[1, 3, 2, 5, 4, 6]$, and the sequence of popped-out elements is $[1, 2, 3, 4, 5, 6]$. What is the minimum possible size of the stack?
  - **Example.** Consider the sequence of pushed-in elements $[1, 3, 2]$, and the sequence of popped-out elements $[1, 2, 3]$. There exists a sequence of stack operations [PUSH, POP, PUSH, PUSH, POP, POP], which means the stack operations are [PUSH(1), POP( ), PUSH(3), PUSH(2), POP( ), POP( )]. During the entire process, the minimum possible number of elements in the stack is 2, so the size of the stack is at least 2.

  Solution: There exists a sequence of stack operations [PUSH(1), POP( ), PUSH(3), PUSH(2),POP( ),POP( )PUSH(5),PUSH(4),POP( ),POP( ),PUSH(6),POP( )]. During the entire process, the minimum possible number of elements in the stack is 2, so the size of the stack is at least 2.
- **(iv). [2 mark]** Assume the sequence of pushed-in elements is $[1, 2, 3, 4, 5]$, and the size of stack is 4. How many different sequences of operations will not result in an error(This means that during the execution of each operation sequence, the number of elements in the stack will not exceed 4)?*HINT: For a stack of unlimited size and*

a sequence of pushed-in elements of length n, the number of different sequences of operations is$\frac{1}{n+1}\binom{2n}{n}$.

* **Example.** Consider the sequence of pushed-in elements $[1, 2, 3]$. If there is no limit to the size of the stack, then it can be directly concluded that the number of different operation sequences is $\frac{1}{3+1}\binom{2\times 3}{3} = 5$. If the size of the stack is 2, then there are 4 possible operation sequences, which are[PUSH, POP, PUSH, POP, PUSH, POP], [PUSH, PUSH, POP, POP, PUSH, POP], [PUSH, POP, PUSH, PUSH, POP, POP] and [PUSH, PUSH, POP, PUSH, POP, POP]respectively, where the number of elements in the stack during each operation sequence will not exceed 2.

Solution: When there is no limit on the size of the stack, the number of different operation sequences is $\frac{1}{5+1}\binom{2\times 5}{5} = 42$. Consider what kind of operation sequence will not meet the limit, that is, the number of elements in the stack reaches 5 during the operation. It can be found that only one sequence of operations [PUSH, PUSH, PUSH, PUSH, PUSH, POP, POP, POP, POP, POP] leads to this situation. So the answer is $42 - 1 = 41$.