CSCE 222 Discrete Structures for Computing – Fall 2023
Hyunyoung Lee

Problem Set 5

Due dates: Electronic submission of *yourLastName-yourFirstName-hw5.tex* and *yourLastName-yourFirstName-hw5.pdf* files of this homework is due on **Monday, 10/23/2023 before 11:59 p.m.** on `https://canvas.tamu.edu`. You will see two separate links to turn in the .tex file and the .pdf file separately. Please do not archive or compress the files. **If any of the two files are missing, you will receive zero points for this homework.**

Name: Kevin Lei **UIN: 432009232**

Resources. (All people, books, articles, web pages, etc. that have been consulted when producing your answers to this homework)

On my honor, as an Aggie, I have neither given nor received any unauthorized aid on any portion of the academic work included in this assignment. Furthermore, I have disclosed all resources (people, books, web sites, etc.) that have been used to answer this homework.

Electronic signature: **Kevin Lei**

Total $100 + 10$ (bonus) points.

The intended formatting is that this first page is a cover page and each problem solved on a new page. You only need to fill in your solution between the `\begin{solution}` and `\end{solution}` environment. Please do not change this overall formatting.

Checklist:
- ☐ Did you type in your name and UIN?
- ☐ Did you disclose all resources that you have used?
  (This includes all people, books, websites, etc. that you have consulted)
- ☐ Did you sign that you followed the Aggie Honor Code?
- ☐ Did you solve all problems?
- ☐ Did you submit both the .tex and .pdf files of your homework to each correct link on Canvas?

Problem 1. (10 points) Section 11.1, Exercise 11.3

Solution. i)

Problem 2. (20 points) Section 11.3, Exercise 11.14. [Requirement: Study the definition of $\asymp$ involving the inequalities carefully and use the definition to answer the questions.]

Solution.

Problem 3. (15 points) Prove that $3n^2 + 41 \in O(n^3)$ by giving a direct proof based on the definition of big-O involving the inequalities and absolute values, as given in the lecture notes Section 11.4.

To do so, first write out what $3n^2 + 41 \in O(n^3)$ means according to the definition. Then, you need to find a positive real constant C and a positive integer n_0 that satisfy the definition.

Solution.

Problem 4. (15 points) Prove that $\frac{1}{2}n^2 + 5 \in \Omega(n)$ by giving a direct proof based on the definition of big-Ω involving the inequalities and absolute values, as given in the lecture notes Section 11.5.

To do so, first write out what $\frac{1}{2}n^2 + 5 \in \Omega(n)$ means according to the definition. Then, you need to find a positive real constant c and a positive integer n_0 that satisfy the definition.

Solution.

Problem 5. (10+10 = 20 points) Read Section 11.6 carefully before attempting this problem.

Analyze the running time of the following algorithm using a step count analysis as shown in the Horner scheme (Example 11.40).

```
// search a key in an array a[1..n] of length n
search(a, n, key)          cost   times
  for k in (1..n) do        c1     [  ]
    if a[k]=key then        c2     [  ]
       return k             c3     [  ]
  endfor                    c4     [  ]
  return false              c5     [  ]
```

(a) Fill in the `[  ]`s in the above code each with a number or an expression involving `n` that expresses the step count for the line of code.

(b) Determine the worst-case complexity of this algorithm and give it in the Θ notation. Show your work and explain using the definition of Θ involving the inequalities.

Solution. (For part (b))

Problem 6. (15+15 = 30 points) Read Section 11.6 carefully before attempting this problem. Analyze the running time of the following algorithm using a step count analysis as shown in the Horner scheme (Example 11.40).

```
// determine the number of digits of an integer n
binary_digits(n)            cost  times
  int cnt = 1                c1    [  ]
  while (n > 1) do           c2    [  ]
    cnt = cnt + 1            c3    [  ]
    n = floor( n/2.0 )       c4    [  ]
  endwhile                   c5    [  ]
  return cnt                 c6    [  ]
```

(a) Fill in the `[  ]`s in the above code each with a number or an expression involving `n` that expresses the step count for the line of code.

(b) Determine the worst-case complexity of this algorithm as a function of n and give it in the Θ notation. Show your work and explain using the definition of Θ involving the inequalities.

Solution. (For part (b))