IFN564 assignment

Released: 30 April 2024
Due date: 19 May 2024

🚨 Note on generative AI tools 🚨

Tools like ChatGPT are part of our world. As a teaching team, we believe that they offer great benefits, but also have weaknesses that any potential user should be aware of.

These tools are already used in the workplace, and some of you may be interested in using them in your studies as well. We think this could be a good learning opportunity.

For this assignment, we are therefore offering you a choice between two paths:

1. A traditional approach where you answer all questions on your own. If you choose this path, no AI tool is allowed.

2. An AI-assisted approach where you are allowed to use ChatGPT or equivalent tools to support your work. In that case, you need to provide the exact prompts you have used, the responses you obtained, and a reflection on whether these responses are appropriate (and why). That reflection is crucial to assess that *you* are meeting the learning outcomes of the unit.

Your submission needs to clearly indicate which path you have taken.

As always, any attempt to portray as your own any work that has been completed by another person or a machine will be treated as a potential case of academic misconduct. If you have any doubts, please get in touch. It is always OK to ask questions.

Imagine that you are working for a software development company, and your client is a shop owner who wants a new system to store customer data.

We will see later in the semester how to handle a customer object. For this assignment, we assume the only information you have about these customers are names, so we treat the collection as a collection of strings. This is not completely realistic of course, but it does not impact the behaviour or efficiency of any the algorithms, which are what the unit is about. For simplicity, we will also assume that all names are unique.

You are trying to decide how to store and manage the customer data.

In the first instance, you are considering using an array to store the customers. For now, we assume that you have already allocated the array to a maximum size N_max, which you hope will be large enough to accommodate all your customers.

You are evaluating two options:
A) You always insert customers at the end, not caring about order.
B) You always keep your structure sorted, by inserting any new customer at their correct position.

This leads you the consider the following algorithms:

Option A

```
Algorithm InsertOptionA(A[0..N_max-1], n, new_customer)
    if n < N_max
        A[n] ← new_customer
        n ← n+1
    else
        print "The array is already full"


Algorithm SearchOptionA(A[0..N_max-1], n, customer)
    for i ← 0 to n − 1 do
        if A[i] = customer
            return i
    return -1


Algorithm DeleteOptionA(A[0..N_max-1], n, customer)
    i ← 0
    while i < n and A[i] ≠ customer do
        i ← i+1
    if i = n
        print "This customer is not in the array"
    else
        while i < n – 1 do
            A[i] ← A[i+1]
            i ← i+1
        n ← n – 1
```

Option B

```
Algorithm InsertOptionB(A[0..N_max-1], n, new_customer)
    if n < N_max
        i ← 0
        while i < n and A[i] < customer do
            i ← i+1
        j ← n – 1
        while j ≥ i do
            A[j+1] ← A[j]
            j ← j – 1
        A[i] ← new_customer
        n ← n+1
    else
        print "The array is already full"

Algorithm SearchOptionB(A[0..N_max-1], n, customer)
    l ← 0; r ← n – 1
    while l ≤ r do
        m ← ⌊(l+r)/2⌋
        if A[m] = customer
            return m
        else if A[m] > customer
            r ← m – 1
        else
            l ← m+1
    return -1

Algorithm DeleteOptionB(A[0..N_max-1], n, customer)
    l ← 0;  r ← n – 1
    while l ≤ r do
        m ← ⌊(l+r)/2⌋
        if A[m] = customer
            while m < n – 1 do
                A[m] ← A[m+1]
                m ← m + 1
            n ← n – 1
            return 0
        else if A[m] > customer
            r ← m – 1
        else
            l ← m+1
    print  "This customer is not in the array"
```

Question 1

Separately for each algorithm in option A, discuss their efficiency. Make sure to follow the steps covered in class and detail your thinking:

1. What is the problem size?
2. What is the basic operation?
3. Does the algorithm have a best case and a worst case?
4. Calculate the exact efficiency function, using a summation formula or recurrence relation as appropriate and showing all steps in the calculation. If the best and worst cases are different, calculate both functions.
5. What is the efficiency class? (or classes, if the best and worst cases are different).

Question 2

Same question, for the algorithms in option B.

Question 3

Which option do you think is best? How much does that depend on the relative number of insertions, deletions, and searches?

Question 4

Implement both options in the programming language of your choice, and run suitable tests to achieve two goals:

1. Ensuring that the algorithms are working as expected.
2. Verifying whether the execution time matches your efficiency analysis from q.1-3.

Make sure to justify the tests you are running and to clearly show and discuss your results.

Question 5

You realise that the number of customers may get quite large, and that it is quite difficult to estimate it. You decide to use a linked list instead of an array.

Without going through the first three questions again in detail, discuss what would change in your approach if using a linked list, and what the impact on efficiency would be.