National University of Computer and Emerging Sciences

Assignment No. 1

Data Structures

CS2001

Fall 2023

Deadline: 18-09-2022, 04:30PM

Department of Computer Science

Submission Instructions:

- All problems must be solved by following the order and submitted as one .zip file.
- Copy all the header (.h) and (.cpp) file in a folder with naming convention given below containing separate subfolders for each question.
    - Parent Folder should be named as DS_ASS01_XXF_YYYY, where XX represents the batch and YYYY represents the roll number. Compress this folder as DS_ASS01_XXF_YYYY.zip and submit this one file.
    - Solution of each question should be in a subfolder, named according to he questions.
- Also submit a .doc file containing the solution of question 01 and screenshots along with the code of rest of the questions.
- Any submission not following the submission instructions will not be evaluated.
- This is an individual assignment.
- **Design proper classes rather than global functions.**
- **Plagiarism is strictly prohibited.**

Q1: For the following program fragment compute the worst-case asymptotic time complexity (as a function of n). Where it says ``loop body'' you can assume that a constant number of lines of code are there. Briefly explain how you obtained your answer. **10**

i)

```
for (i=0; i<=n-1; i++)
{
     for (j=i+1; j<=n-1;j++) {
     loop body
     }
}
```

ii)

```
for(int i =0 ; i < =n ; i++)
{
     for(int j =1; j<= i * i; j++)
     {
          if (j % i == 0)
          {
               for(int k = 0; k<j; k++)
               sum++;
          }
     }
}
```

iii)

```
int a = 0;
for (i = 0; i < N; i++) {
     for (j = N; j > i; j--) {
          a = a + i + j;
     }
}
```

iv)

```
int i, j, k = 0;
for (i = n / 2; i <= n; i++) {
     for (j = 2; j <= n; j = j * 2) {
          k = k + n / 2;
     }
}
```

v)

```
int a = 0, i = N;
while (i > 0) {
     a += i;
     i /= 2;
}
```

Q2. **20**

List ADT Implementation (via dynamic array) Implement the following operations of List ADT by using array class

- Constructors (default, parameterize, copy) & destructor
- void printList ( ) (if empty then program should display proper message)
- int searchElement (int X) (linear and binary search)
- void insertElementAt (int X, int pos) //inserts and element X in the list at a given position
- bool deleteElement (int X) //deletes and element X from the list
- bool isFull ( ) //checks if full or not
- bool isEmpty ( ) // checks if empty or not
- int length ( ) //return the number of elements in the list
- void reverseList ( ) //reverse the list
- void emptyList ( ) //Clears all the list
- void copyList (...) //Copies one list to another list
- Copy array to bigger array if data grows
- Note program should display proper message for each case
- Also write a driver (main) program to test your code (provide menu for all operations). Also find worst case time complexity for insertion and deletion

Note: Use proper classes to hold the functionality of the list ADT and data.

Q3. **10**

Write a program for a company where they need help to calculate the monthly wages of their employees depending upon the number of hours they work in the organization. Your program should be generic for any number of employees. Load the names and working hours of the employees from "Employees.txt" file. Wage Criteria is: (50 x number of hours they've worked in a month)

At the end, show all the employee names and their wages on the screen an also save them in a file namely "Wages.txt".

Note: Use pointers-based implementation of linked list

Q4. **20**

There are **N** people standing in a circle waiting to be executed. The counting out begins at some point in the circle and proceeds around the circle in a fixed direction. In each step, a certain number of people are skipped and the next person is executed. The elimination proceeds around the circle (which is becoming smaller and smaller as the executed people are removed), until only the last person remains, who is given freedom. Given the total number of persons **N** and a number **M** which indicates that **M-1** persons are skipped and **M-th** person is killed in circle. The task is to choose the place in the initial circle so that you are the last one remaining and so survive.

1) Create a circular linked list of size **N**.
2) Traverse through linked list and one by one delete every **M-th** node until there is one node left.
3) Return value of the only left node.

Input : Length of circle : N = 4
Count to choose next : M = 2
Output : 1

Q5. **20**

Adding two polynomials using Linked List

Given two polynomial numbers represented by a linked list. Write a function that add these lists means add the coefficients who have same variable powers. Store the two polynomials in a file and fetch from the file to make the link list. The polynomials could be of any degree.

Example:

Input:

1st number = 5x^3 + 4x^2 + 2x^0
2nd number = 5x^1 + 5x^0

Output:

5x^3 + 4x^2 + 5x^1 + 7x^0

Q6. **20**

Mail System Design using linked list

Hattori is very much inspired by the way GMAIL works. He decides to build his own simple version of GMAIL. He divides the mails into 3 categories, namely: **UNREAD, READ and TRASH.**

UNREAD: The messages that haven't been read.

READ: The messages that is read by the user.

TRASH: The messages deleted by the user.

Now, at any point of time, the user can READ an UNREAD message, Move an UNREAD message to TRASH, move a READ message to TRASH or restore a deleted message back to READ category. Now, Hattori requires your help in determining the messages left in all the categories aligned in the order of their arrival in that category.

Formally: You are given N messages, with ID from 1 to N. Initially all the messages are in the UNREAD section of the mail. Now, Q queries are given in the form as shown below:

Sample Query

Sample Query	Action to be taken
1 X	Move the message with ID X from UNREAD to READ.
2 X	Move the message with ID X from READ to TRASH.
3 X	Move the message with ID X from UNREAD to TRASH.
4 X	Move the message with ID X from TRASH to READ.

Given that all the queries are valid, Help Hattori in Determining the messages in all the 3 sections. The query will be entered by the user.

Input:

This line contains the ID's of the messages. The number of messages can vary.

Output:

First line contains all the space separated message ID'S left in the UNREAD section.

Second line contains all the space separated message ID'S left in the READ section.

Third line contains all the space separated message ID'S left in the TRASH section.

NOTE: In case, any section is empty, print "EMPTY" for that section without double quotes.

Example:

Input:

1 2 3 4 5 6 7 8 9 10

Output:

1 3 4 6 8 10

2 9 5

7

Example output on screen

Initial UNREAD section: 1->2->3->4->5->6->7->8->9->10

READ section : EMPTY

TRASH section : Empty

Query 1 : 1 2

UNREAD section: 1->3->4->5->6->7->8->9->10

READ section : 2
TRASH section : Empty
Query 2 : 1 5
UNREAD section: 1->3->4->6->7->8->9->10
READ section : 2->5
TRASH section : Empty
Query 3 : 3 7
UNREAD section: 1->3->4->6->8->9->10
READ section : 2->5
TRASH section : 7