

Data structure (Software engineering)
Course Code:- CS214



Final Term Project

Please read the instructions carefully:

- 1- Students per project is up to 5 students.**
- 2- Choose one of the two parts partI or PartII.**
- 3- You must deliver ONE file pdf contain your source code**
- 4- On the cover page, write your name(s) in arabic, and IDs in english, Faculty University.**
- 5- Sample test cases are included for each function in each problem.**
- 6- Make sure your programs don't have any errors. It is acceptable to have missing parts of the problem which will cost you the grade of the missing parts, but submitting code with errors will cost you the whole grade.**
- 7- Failing to follow any of the above instructions may severely affect your grade.**
- 8-Using C ++ to implement one of the following research project.**
- 9- for each problem you must write a main function that test your work**
- 9-Your file must named as the following**
Course code-id1-id2-id3-id4-id5

Part I research Library management system

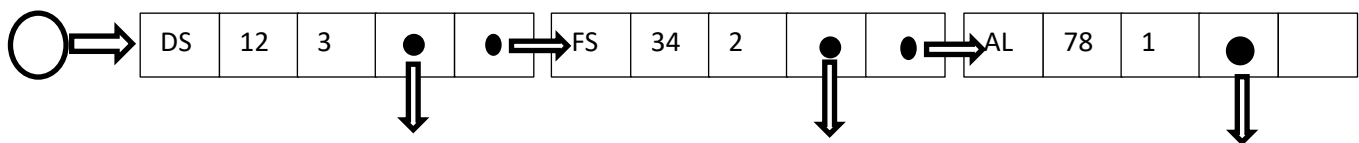
- A. Library management system is a project which aims in developing a system to maintain a specific work of library like adding a new book in book linked list adding book copies in copies queue, Adding book borrow (استعارة), adding book return (إعادة الكتاب المستعار), show borrow statistics (عدد الكتب المستعارة), Library inventory (جرد المكتبة), search about book (البحث في المكتبة عن كتاب معين وتحديد هل هو متاح ام معار)

Features

1. Adding a new book in book linked list, the node of linked list of books consists of the following items

Book name	Book ISBN	Number of copies	Head pointer of the copies queue	Pointer to next book

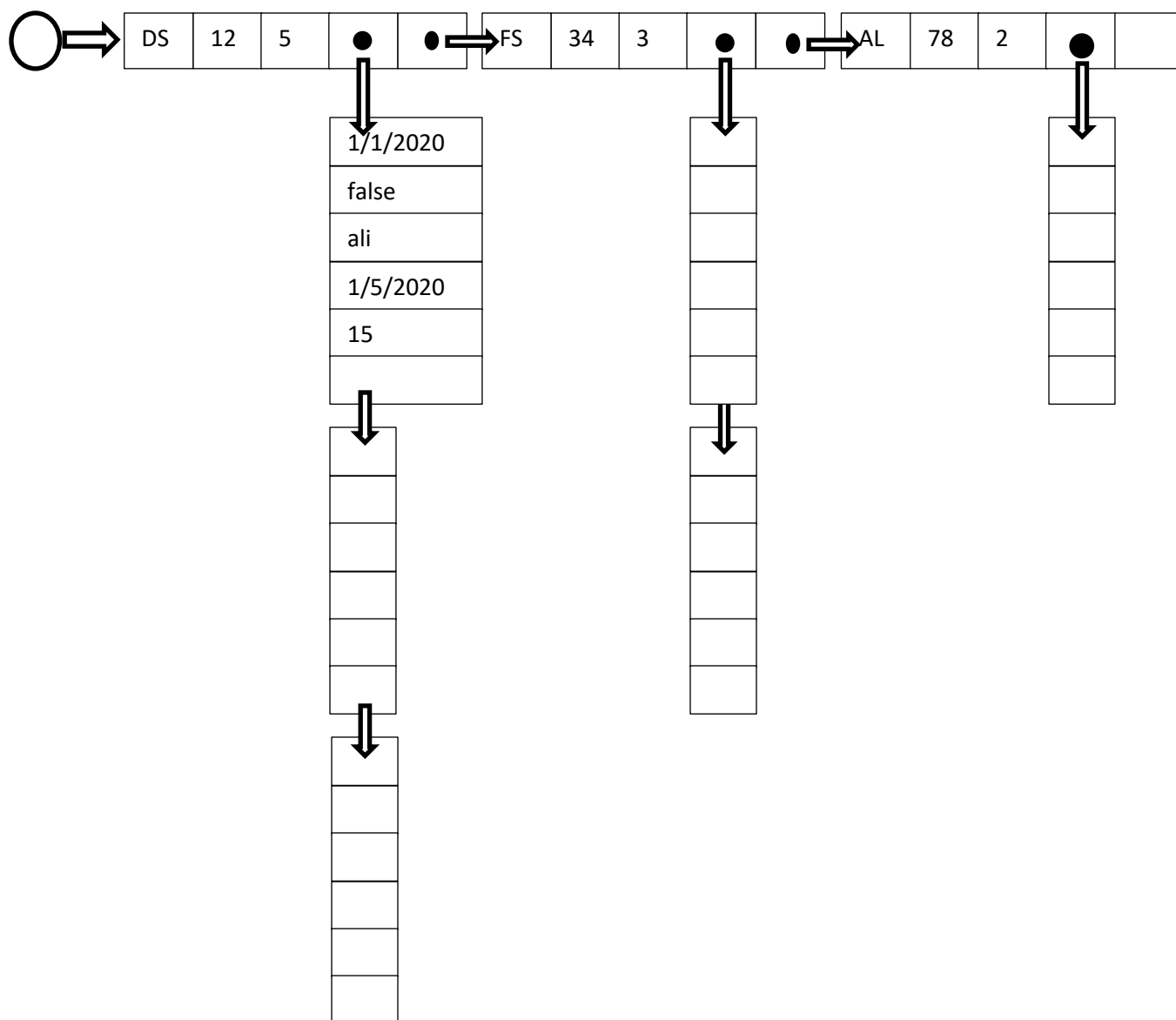
Example after adding three books DS data structure, FS File structure and AL algorithms books.



2. Adding book copies in copies queue, the node of book copies queue consists of the following items

Copy date	Available or is borrow flag	Borrower اسم المستعير	Borrow date	Number of dayes to return the book set it by default 15 days	Pointer to next copy

Example after adding number of copies for each book 3 for DS 2 for FS and 1 for AL be noted that book copy is a queue FIFO



3. Adding book borrow (استعارة)
Update book copies with borrower name and update borrow date and number of days to return the book copy
4. Adding book return (إعادة الكتاب المستعار)
clear borrower and borrow date and number of days to return of the book copy
5. Library inventory (جرد المكتبة)
لكل كتاب عرض عدد النسخ الاجمالية وعدد المستعارة

B. solve the following problems

I. Given the following LinkedList class, main function and expected output:

Linked list class	Main function	Expected output
<pre> template<typename T> struct node { T data; node *next; }; template<typename T> class linked_list { private: node<T> *head, *tail; public: linked_list() { head = NULL; tail = NULL; } }; </pre>	<pre> int main() { linked_list<int> firstLL; firstLL.insert(1); firstLL.insert(20); linked_list<int> secondLL; secondLL.insert(3); secondLL.insert(4); linked_list<int>::concatenate(firstLL.gethead(), secondLL.gethead()); cout << "-- " << endl; linked_list<int>::printLL(firstLL.gethead()); firstLL.deletePos(1); cout << "-- " << endl; linked_list<int>::printLL(firstLL.gethead()); firstLL.deleteVal(20); cout << "-- " << endl; linked_list<int>::printLL(firstLL.gethead()); firstLL.insertPos(30, 1); cout << "-- " << endl; linked_list<int>::printLL(firstLL.gethead()); firstLL.deletePos(10); firstLL.insertPos(40, 11); return 0; } </pre>	<pre> -- 1 20 3 4 -- 1 3 4 -- 1 30 3 4 deletion out of bound Insertion out of bound </pre>

--	--	--

II. It is required to add the following functions to the linked list class:

1. Insert in linked list

Write member insert method in class linked list that takes the value to be inserted as a parameter and inserts the value to the linked list. Make sure to handle insertion in head/tail/middle.

Function signature : `void insert(T n)`

2. Insert in Linked list at certain position

Write member insertPos method in class linked list that takes the value to be inserted and position as parameters and inserts the value to the linked list at the given position. Make sure to handle insertion in head/tail/middle and out of bound cases.

Function signature : `void insertPos(int data, int position)`

3. Delete from linked list

Write member delete method in class linked list that takes the value to be deleted as a parameter and deletes the value from the linked list. Make sure to handle deletion from head/tail/middle.

Function signature : `void deleteVal(T value)`

4. delete from Linked list at certain position

Write member deletePos method in class linked list that takes the value to be deleted and position as parameters and deletes the value from the linked list at the given position. Make sure to handle deletion from head/tail/middle and out of bound cases.

Function signature : `void deletePos(int position)`

5. Concatenate Linked list

Write a static concatenate method that takes the head node of two lists as parameters and adds the second linked list to the first one.

Function signature : `static void concatenate(node<T> *a, node<T> *b)`

6. Print Linked list

Write a static print method that takes the head node as a parameter and prints the linked list recursively

Function signature : `static void printLL(node<T> *head)`

III. Sort Binary array in linear time.

Given a binary array, sort it in linear time and constant space. Output should print contain all zeroes followed by all ones.

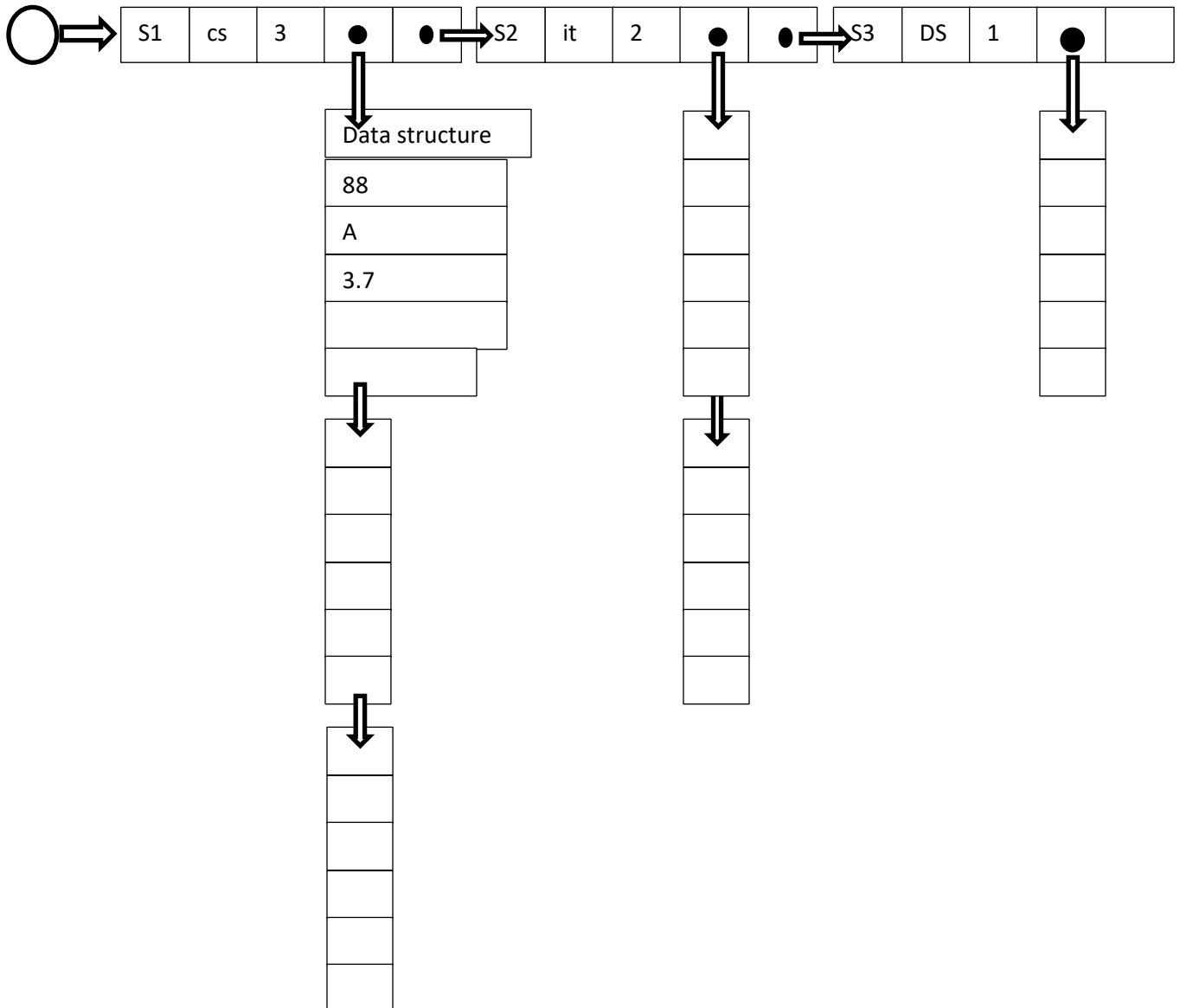
For Example Input:{1,0,1,0,1,0,0,1,0} Output :{0,0,0,0,0,1,1,1,1}

You must solve this problem by three different algorithms .

Part I I research student management system

A. Student management system is a project that have a many function like

- Adding student using linked list
Student linked list items (name , department , number of registered courses , pointer to linked list of students courses , pointer to next students)
- Adding student subject grades as a branched linked list from students which contain (subject name , total ,grade , point)



- Print students subject and grades

B. Solve the following problems

I. In-place merge two sorted arrays

Given two sorted arrays $X[]$ and $Y[]$ of size m and n each, merge elements of $X[]$ with elements of array $Y[]$ by maintaining the sorted order. i.e. fill $X[]$ with first m smallest elements and fill $Y[]$ with remaining elements.

The conversion should be done in-place and without using any other data structure.

For example,

Input:

$X[] = \{ 1, 4, 7, 8, 10 \}$

$Y[] = \{ 2, 3, 9 \}$

Output:

$X[] = \{ 1, 2, 3, 4, 7 \}$

$Y[] = \{ 8, 9, 10 \}$.

II. Assume you have the following Tree Struct

```
template<typename T>
struct Tree {
    Tree(const T &v) : value(v), left(NULL), right(NULL) {}
    T value;
    Tree *left;
    Tree *right;
};
```

1. Print Tree

Write a print method that takes the node which the mirror image of the tree will start from if no parameter is sent to the function the default value will be the root node.

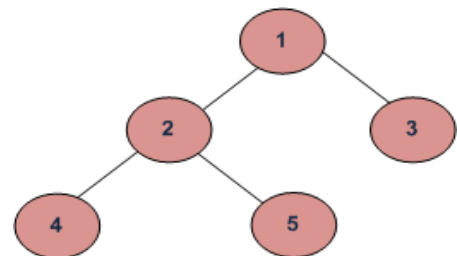
You must solve this problem by three different algorithms.

(Inorder, Preorder and Postorder)

(a) Inorder (Left, Root, Right) : 4 2 5 1 3

(b) Preorder (Root, Left, Right) : 1 2 4 5 3

(c) Postorder (Left, Right, Root) : 4 5 2 3 1



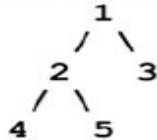
2. Tree Flipping.

Write a flip method that takes the node which the mirror image of the tree will start from if no parameter is sent to the function the default value will be the root node. Flip the tree then print the flipped tree.

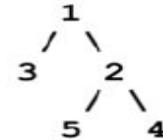
```
void flip(Tree <int>* t);
```

Output: print the flipped tree

Example original tree:



Example new tree:



3. Largest Values

Find the largest value in each row of this tree.

The output should be: [1, 3, 5]

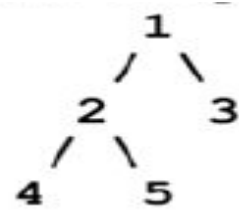
In the first row, there is only one vertex, the root with value 1;

In the second row, there are two vertices with values 2 and 3, so the largest value is 3;

In the third row, there are two vertices with values 4 and 5, so the largest value is 5.

```
void largestValues(Tree <int>* t);
```

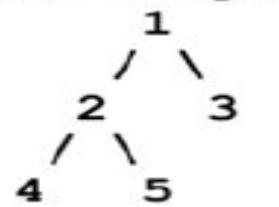
note : - There is many possible answers



4. Branches Sum

Sum encoded values in each path from root to leaf.

```
int main(){
    Tree<int>* t = new Tree<int>(1);
    Tree<int>* _1_left = new Tree<int>(2);
    Tree<int>* _1_right = new Tree<int>(3);
    t->left = _1_left;      t->right = _1_right;
    Tree<int>* _2_left = new Tree<int>(4);
    Tree<int>* _2_right = new Tree<int>(5);
    _1_left->left = _2_left; _1_left->right =
    _2_right;
    largestValues(t);
}
```



The output should be: 262 (124 + 125 + 13)

Path 1 → 2 → 4 encodes 124

Path 1 → 2 → 5 encodes 125

Path 1 → 3 encodes 13

long branchesSum(Tree<int> * t);

III. **Balanced String**

Given a string containing just the characters '(', ')', '{', '}', '[' and ']', determine if the input string is valid.

An input string is valid if:

1. Open brackets must be closed by the same type of brackets.
2. Open brackets must be closed in the correct order.

Note that an empty string is also considered valid.

Input : exp = “{()}[][{(){}O}O]”

Output : “Balanced”

Input : exp = “{(})”

Output : "Not Balanced"