Assignment

Array

Write a single menu driven program containing separate function for each of the following operations for linear array. Ask user to execute functions of choices given in the menu. The function names must be same as below:

- **Insert(item, index):** Insert the given item at the specified index.
- **Delete(index):** delete the item from the specified index.
- **Linear_search(key):** Search the given key in the array using linear search.
- **Binary_search(key):** Search the given key element in the array using binary search.
- **Display():** Print the array elements.

Linear Linked List

Write a single menu driven program containing separate function for each of the following operation on singly linear Linked List. Ask user to execute functions of choice given in the menu. The function names must be same as below:

- **Insert_begin():** insert a new node at the beginning of the list.
- **Insert_last():** insert a new node at the last in the list.
- **Insert_random(pos):** insert a new node at the given position in the list.
- **Insert_specific(key):** insert a new node after the node containing given key.
- **Delete_begin():** delete node from beginning of the list.
- **Delete_end():** delete the last node from the list.
- **Search(key):** search the given key in the list and return the node’s position containing the key.
- **Display():** print all the elements of the list.

Circular Linked List

Write a single menu driven program containing separate function for each of the following operation on singly circular Linked List. Ask user to execute functions of choice given in the menu. The function names must be same as below:

- **Insert_begin():** insert a new node at the beginning of the list.
- **Insert_last():** insert a new node at the last in the list.
- **Insert_random(pos):** insert a new node at the given position in the list.
- **Insert_specific(key):** insert a new node after the node containing given key.
- **Delete_begin():** delete node from beginning of the list.
- **Delete_end():** delete the last node from the list.
- **Search(key):** search the given key in the list and return the node's position containing the key.
- **Display():** print all the elements of the list.

Doubly Linked List

Write a single menu driven program containing separate function for each of the following operation on doubly linear Linked List. Ask user to execute functions of choice given in the menu. The function names must be same as below:

- **Insert_begin():** insert a new node at the beginning of the list.
- **Insert_last():** insert a new node at the last in the list.
- **Insert_random(pos):** insert a new node at the given position in the list.
- **Insert_specific(key):** insert a new node after the node containing given key.
- **Delete_begin():** delete node from beginning of the list.
- **Delete_end():** delete the last node from the list.
- **Search(key):** search the given key in the list and return the node's position containing the key.
- **Display():** print all the elements of the list.

Stack using Array

Implement the stack using array data structure and write a single menu driven program containing separate function for each of the following operation. Ask user to execute any of the functions given in the menu. The function names must be same as below:

- **push(item):** insert an item in the stack.
- **pop():** remove an item from the stack
- **isEmpty():** check the stack is empty or not.
- **isFull():** check the stack is full or not.
- **peek():** return the top item of the stack.
- **display():** Print all the items present in the stack.

Stack using Linked List

Implement the stack using linked list and write a single menu driven program containing separate function for each of the following operation. Ask user to execute any of the functions given in the menu. The function names must be same as below:

- **push(item):** insert an item in the stack.
- **pop():** remove an item from the stack
- **isEmpty():** check the stack is empty or not.
- **isFull():** check the stack is full or not.
- **peek():** return the top item of the stack.
- **display():** Print all the items present in the stack.

Queue using Array

Implement the stack using array and write a single menu driven program containing separate function for each of the following operations on both linear queue and circular queue. Ask user to execute any of the functions given in the menu. The function names must be same as below:

- **Enqueue(item):** Insert the given item in the queue.
- **Dequeue():** Delete an item from the queue.
- **isEmpty():** check the queue is empty or not.
- **isFull():** check the queue is full or not.
- **count():** count the number of elements in the queue.
- **display():** print all the elements of the queue.

Queue using Linked List

Write a single menu driven program containing separate function for each of the following operations on both linear and circular queue. Ask user to execute any of the functions given in the menu. The function names must be same as below:

- **Enqueue(item):** Insert the given item in the queue.
- **Dequeue():** Delete an item from the queue.
- **isEmpty():** check the queue is empty or not.
- **isFull():** check the queue is full or not.
- **count():** count the number of elements in the queue.
- **display():** print all the elements of the queue.