DSA Lab07

23K2001

M.Muzammil Siddiqui

BCS-3J

Q1:

```
//23K2001 Muzammil
#include<iostream>
using namespace std;
class node{
    private:
        int data;
        node* next;
    public:
        node(){next = nullptr;}
        node(int val){
            data = val;
            next = nullptr;
        }

        int getData(){ return data;}
        node* getNext(){return next;}
        void setNext(node* update){next = update;}
};
class singleList{
    private:
        node* head;
        node* tail;
        int size;
    public:
        singleList(){
            head = nullptr;
            tail = nullptr;
            size = 0;
        }
        void display(){
            node* temp = head;
            while(temp!=nullptr)
            {
                cout<<temp->getData()<<"\t";
                temp=temp->getNext();
            }
            cout<<endl;
        }
        void insertAtStart(int val)
        {
            node* n = new node(val);
            n->setNext(head);
```

```
        head = n;
        size++;
    }
    void insertAtEnd(int val)
    {
        node* temp = head;
        node* n = new node(val);
        if(head == NULL){
            head = n;
            tail = n;
        }
        else{
            tail->setNext(n);
            tail = n;
        }
        size++;
    }
    void insertAtIndex(int index,int val){
        node* update = new node(val);
        node* temp = head;
        node* before = nullptr;
        for(int i=0;i<index-1;i++){
            before = temp;
            temp=temp->getNext();
        }
        before->setNext(update);
        update->setNext(temp);
        size++;
    }
    void deleteNode(int val){
        node* before = nullptr;
        node* temp = head;
        while(temp->getData()!=val){
            before = temp;
            temp = temp->getNext();
        }
        before->setNext(temp->getNext());
        delete temp;
        size--;
    }
    void reverseMN(int m,int n){
        if(head==nullptr || m < 1 || n > size || m >= n){
            cout<<"Error applying function!"<<endl;
            return;
        }
```

```
            node* current = head;
            node* before = nullptr;
            node* newTail = nullptr;

            for(int i = 1;i<m;i++){
                before = current;
                current = current->getNext();
            }
            newTail = current;
            node* next = nullptr;
            for(int i = m;i<=n;i++){
                next = current->getNext();
                current->setNext(before);
                before = current;
                current = next;
            }
            if(before!=nullptr){
                if(m == 1)
                    head = before;
                else
                {
                    node* temp = head;
                    for (int i = 1; i < m - 1; i++)
                        temp = temp->getNext();
                    temp->setNext(before);
                }
            }
            newTail->setNext(current);
        }

};
int main(){
    singleList flex;
    cout<<"How many elements: ";
    int e,v;
    cin>>e;
    cout<<"Enter "<<e<<" elements: ";
    for(int i=0;i<e;i++){
        cin>>v;
        flex.insertAtEnd(v);
    }
    cout<<endl<<"your List:"<<endl;
    flex.display();

    cout<<"Enter indexes MxN to reverse: ";
```

```
    int m,n;
    cin>>m>>n;
    cout<<endl<<"After reversing from M to N:"<<endl;
    flex.reverseMN(m,n);
    flex.display();
    return 0;
}
```

```
How many elements: 7
Enter 7 elements: 10 20 30 40 50 60 70

your List:
10      20      30      40      50      60      70
Enter indexes MxN to reverse: 3 6

After reversing from M to N:
10      20      60      50      40      30      70
PS F:\Semester Material - Muzammil\FAST-KHI-Semester-3\Data Structures (LAB)\Lab Tasks
```

```
How many elements: 7
Enter 7 elements: 10 20 30 40 50 60 70

your List:
10      20      30      40      50      60      70
Enter indexes MxN to reverse: 1 4

After reversing from M to N:
40      30      20      10      50      60      70
PS F:\Semester Material - Muzammil\FAST-KHI-Semester-3\Data Structures (LAB)\Lab Tasks\
```

```
How many elements: 7
Enter 7 elements: 10 20 30 40 50 60 70

your List:
10      20      30      40      50      60      70
Enter indexes MxN to reverse: 2 1

After reversing from M to N:
Error applying function!
10      20      30      40      50      60      70
PS F:\Semester Material - Muzammil\FAST-KHI-Semester-3\Data Structures (LAB)\Lab Tasks\
```

Q2:

```
//23K2001 Muzammil
#include<iostream>
using namespace std;
class node{
    private:
        int data;
        node* next;
    public:
        node(){next = nullptr;}
        node(int val){
            data = val;
            next = nullptr;
        }

        int getData(){ return data;}
        node* getNext(){return next;}
        void setNext(node* update){next = update;}
};
class singleList{
    private:
        node* head;
        node* tail;
        int size;
    public:
        singleList(){
            head = nullptr;
            tail = nullptr;
            size = 0;
        }
        void display(){
            node* temp = head;
            while(temp!=nullptr)
            {
                cout<<temp->getData()<<"\t";
                temp=temp->getNext();
            }
            cout<<endl;
        }

        void insertAtStart(int val)
        {
            node* n = new node(val);
```

```
        n->setNext(head);
        head = n;
        size++;
    }
    void insertAtEnd(int val)
    {
        node* temp = head;
        node* n = new node(val);
        if(head == NULL){
            head = n;
            tail = n;
        }
        else{
            tail->setNext(n);
            tail = n;
        }
        size++;
    }
    void insertAtIndex(int index,int val){
        node* update = new node(val);
        node* temp = head;
        node* before = nullptr;
        for(int i=0;i<index-1;i++){
            before = temp;
            temp=temp->getNext();
        }
        before->setNext(update);
        update->setNext(temp);
        size++;
    }
    void deleteNode(int val){
        node* before = nullptr;
        node* temp = head;
        while(temp->getData()!=val){
            before = temp;
            temp = temp->getNext();
        }
        before->setNext(temp->getNext());
        delete temp;
        size--;
    }
    void removeDup(){
    if(head==nullptr)
        return;
    node* current = head;
```

```
        while (current && current->getNext()) {
            if (current->getData() == current->getNext()->getData()) {
                node* temp = current->getNext();
                current->setNext(current->getNext()->getNext());
                delete temp;
                size--;
            }
            else
                current = current->getNext();
        }
        tail = current;
    }

};
int main(){
    singleList flex;
    cout<<"How many elements: ";
    int e,v;
    cin>>e;
    cout<<"Enter "<<e<<" elements: ";
    for(int i=0;i<e;i++){
        cin>>v;
        flex.insertAtEnd(v);
    }
    cout<<endl<<"your List:"<<endl;
    flex.display();

    cout<<endl<<"After removing duplicates:"<<endl;
    flex.removeDup();
    flex.display();
    return 0;
}
```

```
How many elements: 4
Enter 4 elements: 2 2 4 5

your List:
2       2       4       5

After removing duplicates:
2       4       5
PS F:\Semester Material - Muzammil\
```

```
How many elements: 6
Enter 6 elements: 2 2 2 2 3 3

your List:
2       2       2       2       3       3

After removing duplicates:
2       3
PS F:\Semester Material - Muzammil\FAST-KHI-Semester-3\
```

Q3:

```cpp
//23K2001 - Muzammil
#include <iostream>
using namespace std;

void swap(int &a, int &b){
    int temp = a;
    a = b;
    b = temp;
}
void sort(int *arr1,int *arr2,int s1,int s2){
    int index = 0;
    for(int i=0;i<s2;i++){
        int val = arr2[i];
        for (int j=0;j<s1;j++){
            if(val == arr1[j]){
                swap(arr1[index], arr1[j]);
                index++;
            }
        }
    }
}

int main() {
    int n,m;
    cout<<"Enter length of Array#1: ";
    cin>>n;
    int arr1[n];
    cout<<"Input "<<n<<" elements:"<<endl;
    for(int i=0;i<n;i++)
        cin>>arr1[i];
    cout<<"Enter length of Array#2: ";
    cin>>m;
    int arr2[m];
    cout<<"Input "<<m<<" elements:"<<endl;
    for(int i=0;i<m;i++)
        cin>>arr2[i];

    cout<<"Array#1:"<<endl;
    for(int i:arr1)
        cout<<i<<"\t";
    cout<<endl<<"Array#2:"<<endl;
    for(int i:arr2)
```

```
        cout<<i<<"\t";
    cout<<endl<<endl<<"Array#1 after sorting to Array#2:"<<endl;
    sort(arr1,arr2,n,m);
    for(int i:arr1)
        cout<<i<<"\t";
    return 0;
}
```

```
Enter length of Array#1: 6
Input 6 elements:
7 13 4 7 10 4
Enter length of Array#2: 2
Input 2 elements:
7 4
Array#1:
7       13      4       7       10      4
Array#2:
7       4

Array#1 after sorting to Array#2:
7       7       4       4       10      13
PS F:\Semester Material - Muzammil\FAST-KHI-Semester-3\
```

Q4:

```
//23K2001 - Muzammil
#include <iostream>
using namespace std;

class stacks{
    private:
        int top,size;
    public:
        int *arr;
        stacks():top(-1),size(0),arr(nullptr){}
        stacks(int s):top(-1),size(s){
            arr = new int[size];
            for(int i=0;i<size;i++)
                arr[i]=0;
        }
        int getSize(){ return size; }
        void push(int e){
            if(top>=(size-1)){
                cout<<"Stack overflow occured!"<<endl;
                return;
            }
            arr[++top] = e;
        }
        int pop(){
            if(top<0){
                cout<<"Stacks underflow occured!"<<endl;
                return -1;
            }
            int last = arr[top--];
            return last;
        }
        int peek(){
            if(top < 0){
                cout<<"Stack is Empty";
                return 0;
            } else{
                int x = arr[top];
                return x;
            }
        }
        void display(){
            if(isEmpty()){
```

```
                cout<<"Stack is empty!"<<endl;
                return;
            }
            for(int i=0;i<=top;i++)
                cout<<arr[i]<<" ";
            cout<<endl;
        }
        bool isEmpty(){ return (top<0); }
        ~stacks(){ delete[] arr; }
};

class queue{
    int *arr;
    int size;
    int rear;
    int front;
public:
    queue():front(-1),rear(-1),size(0),arr(nullptr){}
        queue(int s):front(-1),rear(-1),size(s){
            arr = new int[size];
            for(int i=0;i<size;i++)
                arr[i]=-1;
        }
        void enqueue(int q){
            if(isFull()){
                cout<<"Queue is full!"<<endl;
                return;
            }
            else if(isEmpty())
                front = rear = 0;
            else
                rear++;
            arr[rear]=q;
        }
        void dequeue(){
            if(isEmpty()){
                cout<<"Queue is empty!"<<endl;
                return;
            }
            else if(front == rear)
                front = rear = -1;
            else
                front++;
        }
        bool isEmpty(){
```

```
            if(front==-1 || front>rear)
                return true;
            return false;
        }
        bool isFull(){
            if(rear==size-1)
                return true;
            return false;
        }
        int peek(){ return arr[front]; }

        ~queue(){ delete[] arr; }
};
void reverseByQueue(stacks &s){
    queue q(s.getSize());
    while (!s.isEmpty()){
        q.enqueue(s.peek());
        s.pop();
    }

    while (!q.isEmpty()){
        s.push(q.peek());
        q.dequeue();
    }
}
int main()
{
    int s;
    cout<<"Enter size of stack: ";
    cin>>s;
    stacks flex(s);
    int e;
    cout<<"Enter "<<s<<" elements:"<<endl;
    for(int i=0;i<s;i++){
        cin>>e;
        flex.push(e);
    }
    cout<<"Elements in stack: "<<endl;
    flex.display();
    reverseByQueue(flex);
    cout<<endl<<"After reversing using queue:"<<endl;
    flex.display();
}
```

```
Enter size of stack: 4
Enter 4 elements:
10 20 30 40
Elements in stack:
10 20 30 40

After reversing using queue:
40 30 20 10
PS F:\Semester Material - Muzammil\FAST-KHI-Semester-3\
rial - Muzammil\FAST-KHI-Semester-3\Data Structures (LA
3K2001 } ; if ($?) { .\Q4_23K2001 }
Enter size of stack: 3
Enter 3 elements:
6 5 4
Elements in stack:
6 5 4

After reversing using queue:
4 5 6
PS F:\Semester Material - Muzammil\FAST-KHI-Semester-3\
```

```
Enter size of stack: 6
Enter 6 elements:
2 3 2 0 0 1
Elements in stack:
2 3 2 0 0 1

After reversing using queue:
1 0 0 2 3 2
PS F:\Semester Material - Muzammil\FAST-KHI-Semester-3\Data Structures (LAB)
```

Q5:

```
//23K2001 - Muzammil
#include <iostream>
using namespace std;

class stacks{
    private:
        int top,size;
    public:
        int *arr;
        stacks():top(-1),size(0),arr(nullptr){}
        stacks(int s):top(-1),size(s){
            arr = new int[size];
            for(int i=0;i<size;i++)
                arr[i]=0;
        }
        int getSize(){ return size; }
        void push(int e){
            if(top>=(size-1)){
                cout<<"Stack overflow occured!"<<endl;
                return;
            }
            arr[++top] = e;
        }
        int pop(){
            if(top<0){
                cout<<"Stacks underflow occured!"<<endl;
                return -1;
            }
            int last = arr[top--];
            return last;
        }
        int peek(){
            if(top < 0){
                cout<<"Stack is Empty";
                return 0;
            } else{
                int x = arr[top];
                return x;
            }
        }
        void display(){
            if(isEmpty()){
```

```
                cout<<"Stack is empty!"<<endl;
                return;
            }
            for(int i=0;i<=top;i++)
                cout<<arr[i]<<" ";
            cout<<endl;
        }
        bool isEmpty(){ return (top<0); }
        void deleteMiddle(int count=0){
            if (count == (size)/2){
                cout<<endl<<"Middle element: "<<pop()<<endl;
                return;
            }
            int val = pop();
            deleteMiddle(count + 1);
            push(val);
        }

        ~stacks(){ delete[] arr; }
};

int main(){
    int s;
    cout<<"Enter size of stack: ";
    cin>>s;
    stacks flex(s);
    int e;
    cout<<"Enter "<<s<<" elements:"<<endl;
    for(int i=0;i<s;i++){
        cin>>e;
        flex.push(e);
    }
    cout<<"Elements in stack: "<<endl;
    flex.display();
    flex.deleteMiddle();
    cout<<endl<<"After deleting middle element:"<<endl;
    flex.display();
}
```

```
Enter size of stack: 5
Enter 5 elements:
1 2 3 4 5
Elements in stack:
1 2 3 4 5

Middle element: 3

After deleting middle element:
1 2 4 5
PS F:\Semester Material - Muzammil\FAST-KHI-Semester-3\Data Structures (LAB)\
```

```
Enter size of stack: 6
Enter 6 elements:
1 2 3 4 5 6
Elements in stack:
1 2 3 4 5 6

Middle element: 3

After deleting middle element:
1 2 4 5 6
PS F:\Semester Material - Muzammil\FAST-KHI-Semester-3\Data Structures (LAB)\
```

```
Enter size of stack: 7
Enter 7 elements:
2 3 2 0 0 0 1
Elements in stack:
2 3 2 0 0 0 1

Middle element: 0

After deleting middle element:
2 3 2 0 0 1
PS F:\Semester Material - Muzammil\FAST-KHI-Semester-3\Data Structures (LAB)\
```

Q6:

```
//23K2001 - Muzammil
#include<iostream>
using namespace std;
class stacks{
    private:
        int top,size;
    public:
        char *arr;
        stacks():top(-1),size(0),arr(nullptr){}
        stacks(int s):top(-1),size(s){
            arr = new char[size];
            for(int i=0;i<size;i++)
                arr[i]='!';
        }
        int getpeek(){ return top; }
        int getSize() { return size; }

        void push(char e){
            if(top>=(size-1)){
                cout<<"Stack overflow occured!"<<endl;
                return;
            }
            arr[++top] = e;
        }
        char pop(){
            if(top<0){
                cout<<"Stacks underflow occured!"<<endl;
                return '!';
            }
            char last = arr[top--];
            return last;
        }
        char peek(){
            if(top < 0){
                cout << "Stack is Empty";
                return 0;
            } else{
                int x = arr[top];
                return x;
            }
        }
        bool isEmpty(){ return (top<0); }
```

```
};
int precedence(char c){
    if(c == '^')
        return 3;
    else if(c == '*' || c == '/')
        return 2;
    else if(c == '+' || c == '-')
        return 1;
    else
        return -1;
}
string infixToPostfix(string infix){
    string postfix = "";
    stacks s(infix.length());

    for(int i = 0; i < infix.length(); i++){
        char c = infix[i];

        if((c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z'))
            postfix += c;
        else if(c == '(')
            s.push(c);
        else if(c == ')'){
            while(!s.isEmpty() && s.peek() != '('){
                char op = s.pop();
                postfix += op;
            }
            if(s.peek() == '(')
                s.pop();
        } else{
            while(!s.isEmpty() && precedence(c) <= precedence(s.peek())){
                char op = s.pop();
                postfix += op;
            }
            s.push(c);
        }
    }
    while(!s.isEmpty()){
        char op = s.pop();
        postfix += op;
    }

    return postfix;
}
int main(){
```

```
    cout<<"\t\tWelcome to Infix to Postfix expression converter!"<<endl;
    cout<<endl<<"Enter you infix expression: "<<endl;
    string e;
    cin>>e;
    cout<<endl<<"After conversion :-"<<endl;
    cout<<"Postfix: "<<infixToPostfix(e)<<endl;
    // “((A + B) – C * (D / E)) + F”
    return 0;
}
```

```
                Welcome to Infix to Postfix expression converter!

Enter you infix expression:
((A+B)-C*(D/E))+F

After conversion :-
Postfix: AB+CDE/*-F+
PS F:\Semester Material - Muzammil\FAST-KHI-Semester-3\Data Structures (LAB)\Lab Tasks\Lab07
```

```
                Welcome to Infix to Postfix expression converter!

Enter you infix expression:
((a+b)-c*(d/e))+f

After conversion :-
Postfix: ab+cde/*-f+
PS F:\Semester Material - Muzammil\FAST-KHI-Semester-3\Data Structures (LAB)\Lab Tasks\Lab07
```

Q7:

```cpp
//23K2001 - Muzammil
#include <iostream>
using namespace std;

void print(int i,int upto){
    if(i<=upto){
        if(i%2==0)
            cout<<i-1;
        else
            cout<<i+1;
    }
    else
        return;
    i++;
    cout<<"\t";
    print(i,upto);
}

int main(){
    print(1,10);
    return 0;
}
```

```
2       1       4       3       6       5       8       7       10      9
PS F:\Semester Material - Muzammil\FAST-KHI-Semester-3\Data Structures (LAB)
```

Q8:

```cpp
//23K2001 - Muzammil
#include <iostream>
using namespace std;

int seqA(int upto){
    if(upto==1)
        return 1;

    return upto+seqA(upto-1);
}
int seqB(int x){
    if(x==0 || x==1)
        return x;

    return seqB(x-1)+seqB(x-2);
}

int main() {
    cout<<"Sequence A:"<<endl;
    seqA(10);
    for(int i=1;i<=10;i++)
        cout<<seqA(i)<<"\t";
    cout<<endl<<endl;
    cout<<"Sequence B:"<<endl;
    for(int i=0;i<10;i++)
        cout<<seqB(i)<<"\t";
    return 0;
}
```

```
Sequence A:
1       3       6       10      15      21      28      36      45      55

Sequence B:
0       1       1       2       3       5       8       13      21      34
PS F:\Semester Material - Muzammil\FAST-KHI-Semester-3\Data Structures (LAB)\
```

Q9:

```cpp
//23K2001 - Muzammil
#include <iostream>
using namespace std;

bool checkPrime(int x,int i=2){
    if(x<2)
        return false;
    if(i*i>x)
        return true;
    if(x%i==0)
        return false;

    return checkPrime(x,i+1);
}
void composite_primes(int m, int n) {
    if(m>n) return;
    if(checkPrime(m))
        cout<<m<<"(P)"<<endl;
    else if(m>1)
        cout<<m<<"(C)"<<endl;
    else
        cout<<m<<"(Neither)"<<endl;
    composite_primes(m + 1, n);
}
int main() {
    composite_primes(0,15);
    return 0;
}
```

```
0(Neither)
1(Neither)
2(P)
3(P)
4(C)
5(P)
6(C)
7(P)
8(C)
9(C)
10(C)
11(P)
12(C)
13(P)
14(C)
15(C)
PS F:\Semester Material - Muzammil\FAST-KHI-Semester-3\Data Structures
```