ASSIGNMENT-5

Name: Prabhnoor Singh Roll No: 102115059 Group:2NC3

Q1. Given array A[] with sliding window of size w which is moving from the very left of the array to the very right.

Assume that we can only see the w numbers in the window. Each time the sliding window moves rightwards by one position. For example: The array is [1 3 -1 -3 5 3 6 7], and w is 3.

#include <iostream> using namespace std;

void slidingWindow(int arr[],int n,int w){ int max,c=-1; while(c+w<n){ max=c+1;

for(int i=c+1;i<=c+w;i++){ cout<<arr[i]<<", "; if(arr[max]<arr[i]) max=i;

}

cout<<" Max="<<arr[max]<<endl;

c=c+1;

}

}

int main(){

int n;

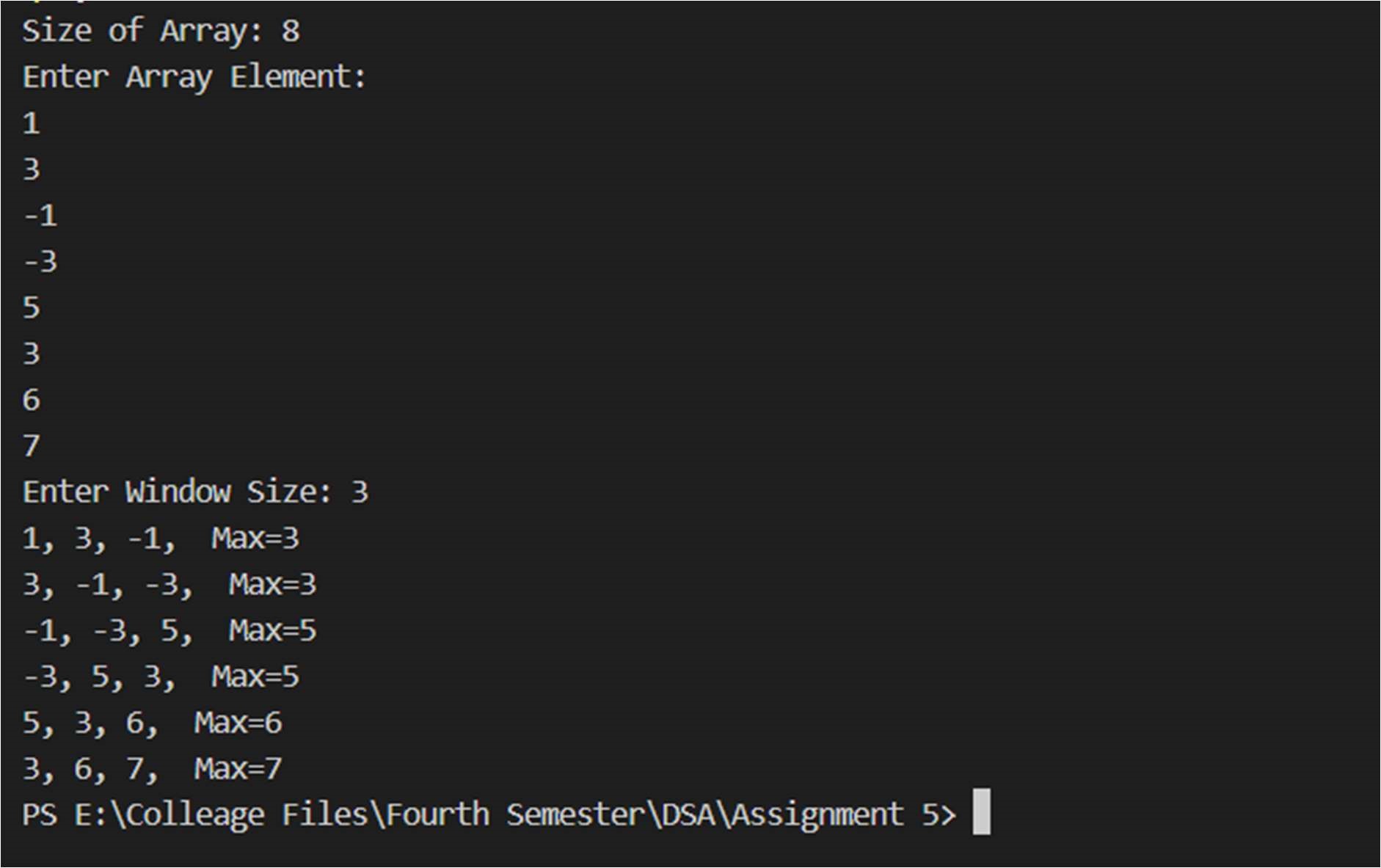
cout<<"Size of Array: "; cin>>n; int arr[n],w;

cout<<"Enter Array Element: "<<endl; for(int i=0;i<n;i++) cin>>arr[i];

cout<<"Enter Window Size: "; cin>>w;

slidingWindow(arr,n,w); return 0;

}



Q2 Given a Linked list, List1 = {A1 , A2 , . . . An–1 ; An) with data, write a program to re-order it to {A1 , An , A2 , An–1 …} without using any extra space.

#include <iostream> using namespace std; struct Node { int data; struct Node\* next;

};

Node\* newNode(int key){ Node\* temp = new Node; temp->data = key; temp->next = NULL; return temp;

}

void reverselist(Node\*\* head){

Node \*prev = NULL, \*curr = \*head, \*next; while (curr) { next = curr->next; curr->next = prev; prev = curr; curr = next;

}

\*head = prev;

}

void printlist(Node\* head){ while (head != NULL) { cout << head->data << " "; if (head->next) cout << "-> "; head = head->next;

}

cout << endl;

}

void rearrange(Node\*\* head){ Node \*slow = \*head, \*fast = slow->next; while (fast && fast->next) { slow = slow->next; fast = fast->next->next;

}

Node\* head1 = \*head; Node\* head2 = slow->next; slow->next = NULL; reverselist(&head2);

\*head = newNode(0); Node\* curr = \*head; while (head1 || head2) { if (head1) { curr->next = head1; curr = curr->next; head1 = head1->next;

}

if (head2) { curr->next = head2; curr = curr->next; head2 = head2->next;

}

}

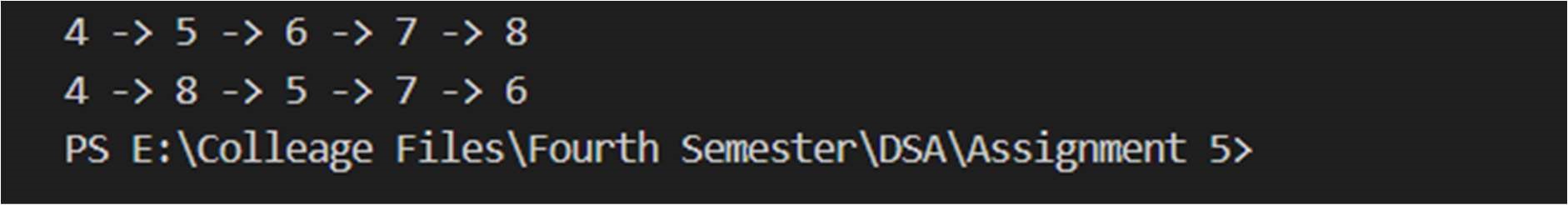
\*head = (\*head)->next;

}

int main(){

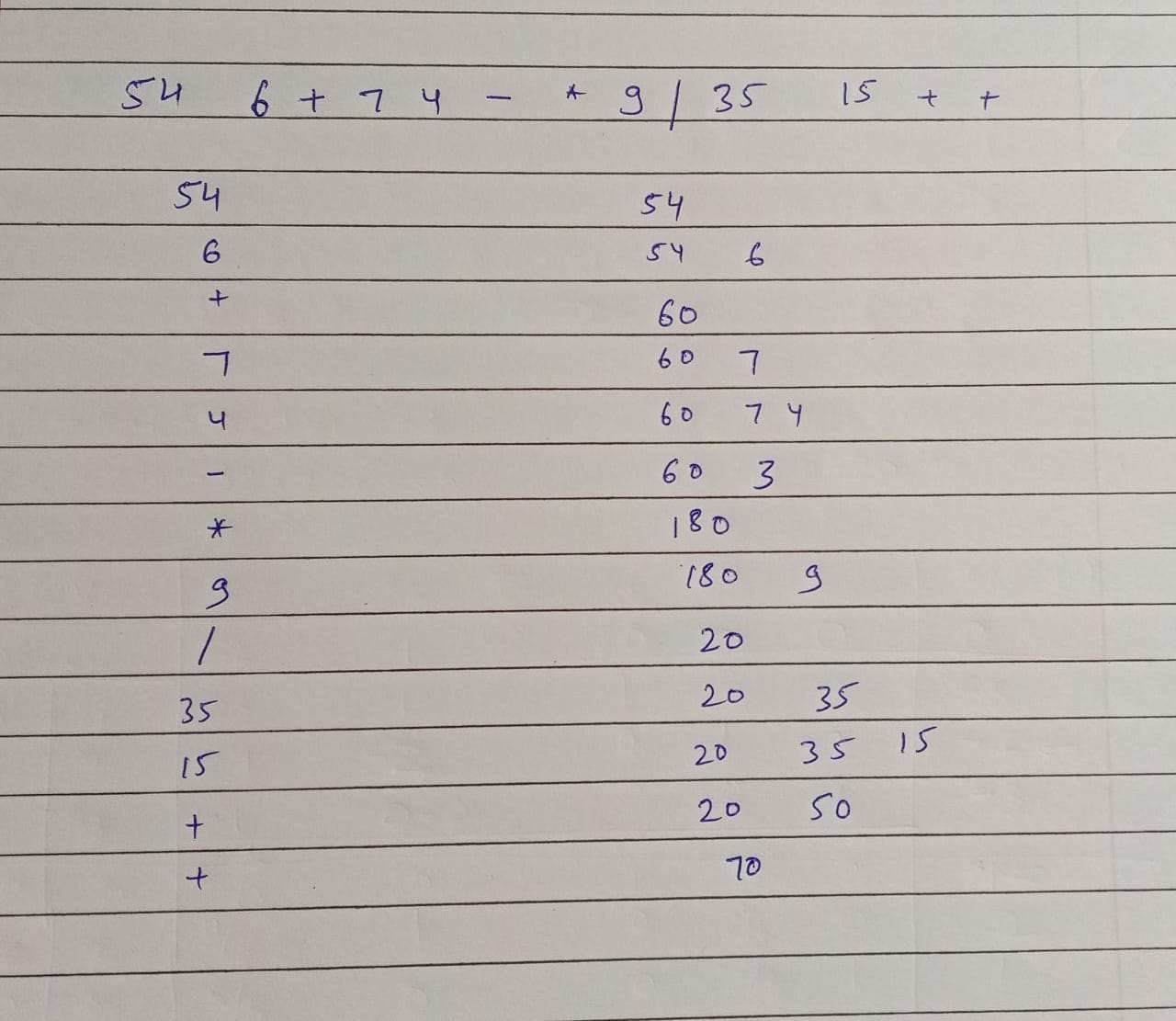
Node\* head = newNode(4); head->next = newNode(5); head->next->next = newNode(6); head->next->next->next = newNode(7); head->next->next->next->next = newNode(8); printlist(head); rearrange(&head); printlist(head); return 0;

}



Q3

❖ What is the value of the following postfix expression? 54 6 + 7 4 - \* 9 / 35 15 + +



#include <bits/stdc++.h> using namespace std;

int prec(char ch){ if(ch=='+'||ch=='-') return 1; else if(ch=='\*'||ch=='/') return 2; else if(ch=='^') return 3; else return -1;

}

string infToPost(string inf){ string res; stack<char> ch; for (int i=0;i<inf.length();i++){ char c = inf[i];

if((c>='a'&&c<='z')||(c>='A'&&c<='Z')||(c>='0'&&c<='9')) res = res + c; else if(c=='(') ch.push(c); else if (c==')'){ while(ch.top()!='('){ res = res + ch.top(); ch.pop();

}

ch.pop();

} else{

while (!ch.empty() and prec(ch.top()) >= prec(c)){ res=res+ch.top(); ch.pop();

}

ch.push(c);

}

}

while (!ch.empty()){ res=res+ch.top(); ch.pop();

}

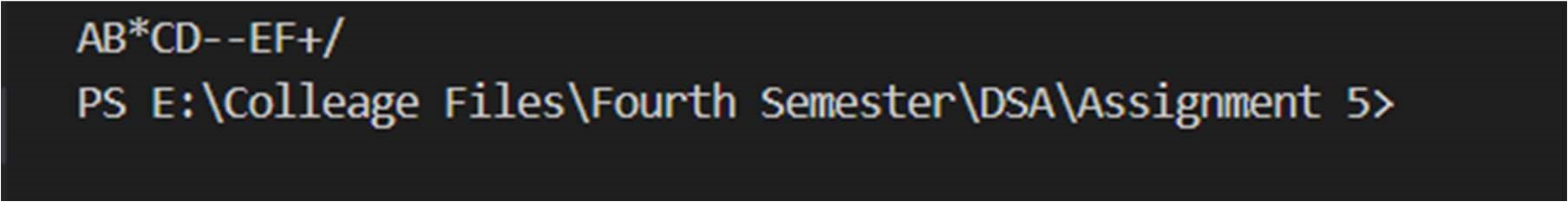
return res;

}

int main(){

string exp = "(A\*B-(C-D))/(E+F)"; cout << infToPost(exp); return 0;

}



Q4. Write a program to perform Parenthesis matching in an expression

#include <bits/stdc++.h> using namespace std; bool check(string exp){ stack<char> ch; int flag;

for(int i=0;i<exp.length();i++){ if(exp[i]=='{'||exp[i]=='['||exp[i]=='(') ch.push(exp[i]); else{

if (!ch.empty()){

if(exp[i]=='}'&&ch.top()=='{'||exp[i]==']'&&ch.top()=='['||ex p[i]==')'&&ch.top()=='(')

ch.pop(); else return false;

} else return false;

}

}

if(ch.empty()) return true;

return false;

}

int main(){ string exp; cout<<"Enter Exp: "; cin>>exp; if (check(exp)==0)

cout<<"Expression is not balanced"<<endl; else

cout<<"Expression is balanced!"<<endl; return 0;

}

