**Logo, company name

Description automatically generated**

**Department of IT and Computer Science**

**Pak-Austria Fachhochschule: Institute of Applied Sciences and Technology, Haripur, Pakistan**

**COMP-201L Data Structures and Algorithms Lab**

**Lab Report 07**

**Class: Computer Science**

**Name: Yaseen Ejaz Ahmed**

**Registration No.: B20F0283CS014**

**Semester: Third**

**Submitted to: Engr. Rafi Ullah**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Instructor Signature**

**Lab No. 7**

**Queues**

**Objectives:**

To learn about Queues and its applications using Arrays and Linked Lists

**Tools/Software Required:**

C++ Compiler

**Introduction:**

A queue is an ordered collection of items where the addition of new items happens at one end, called the “rear,” and the removal of existing items occurs at the other end, commonly called the “front.” As an element enters the queue it starts at the rear and makes its way toward the front, waiting until that time when it is the next element to be removed.

**Lab Tasks:**

**Lab Task 01:** Enter a name character by character in a queue. Your program should Enqueue, Dequeue, and have flags to show whether the queue is empty or full. ASCII value of dequeue should also be displayed.

**USING LINKED LIST:**

**Code:**

#include <iostream>

using namespace std;

**class node**

{

**private:**

char data;

node\* next;

node\* head=NULL;

node\* ptr;

node\* tail;

**public:**

**void Enqueue(char s)**

{

node\*temp=new node();

temp->data=s;

temp->next=NULL;

if(head==NULL)

{

head=temp;

ptr=head;

}

else

{

ptr->next=temp;

ptr=ptr->next;

tail=ptr;

}

}

**int Dequeue()**

{

node\*temp=head;

int ASCII;

ASCII=int(temp->data);

head=head->next;

delete temp;

return ASCII;

}

**bool isEmpty()**

{

if(head==NULL)

return true;

return false;

}

**int show()**

{

if(head!=NULL)

{

cout<<"\nThe values in the linked list are :\n\n";

ptr=head;

while(ptr->next!=NULL)

{

cout<<ptr->data;

ptr=ptr->next;

}

cout<<ptr->data;

cout<<"\n-------------------------------------------------------------";

}

else return 0;

}

};

**int main()**

{

node n;

int opt;

char s;

cout<<"\t\t\tNAMES";

do

{

cout<<"\n\nWould you like to:\n1. Add a character\n2. Remove a character\n3. Exit\n";

cin>>opt;

if(opt==1)

{

cout<<"\n\nEnter a name : ";

cin>>s;

n.Enqueue(s);

n.show();

}

else if(opt==2)

{

if(!n.isEmpty())

{

cout<<"\nThe ASCII of letter removed is "<<n.Dequeue();

}

else

{

cout<<"\n\nTHE QUEUE IS EMPTY";

}

n.show();

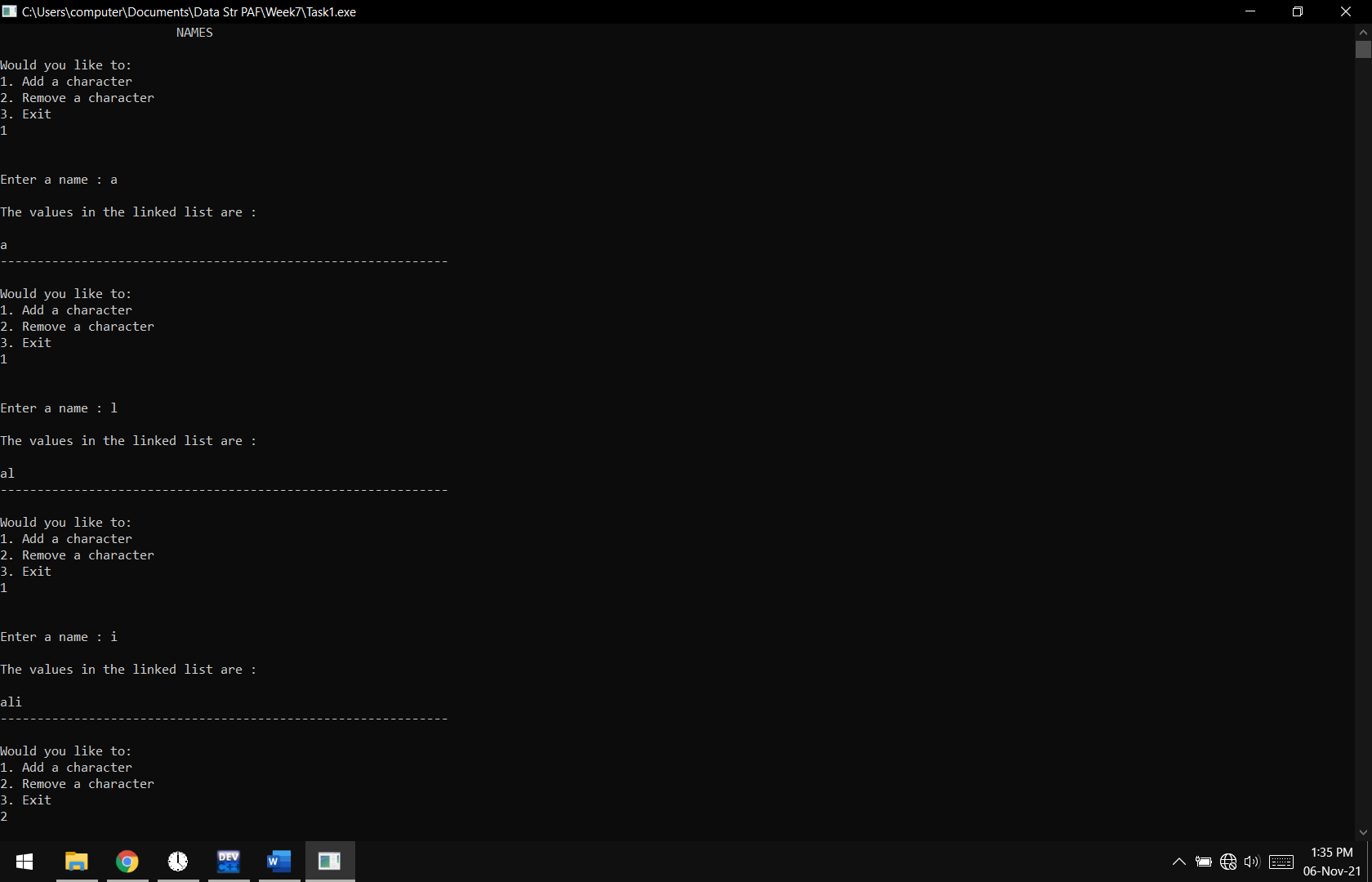
}

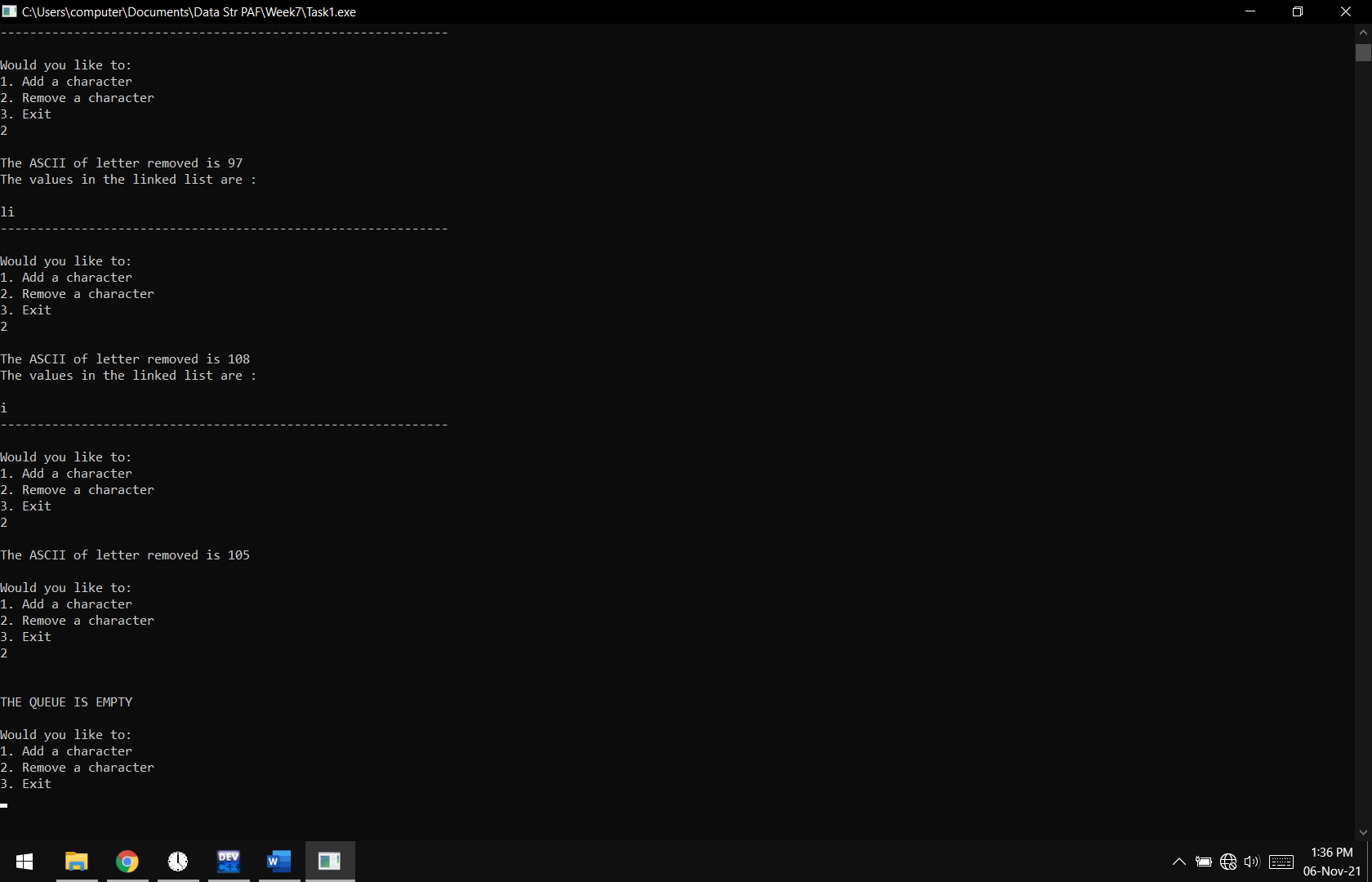
}

while(opt!=3);

}

**Output:**





**Lab Task 02:** Enter a name character by character in a queue. Your program should Enqueue, Dequeue, and have flags to show whether the queue is empty or full. ASCII value of dequeue should also be displayed.

**USING ARRAY:**

**Code:**

#include <iostream>

const int size=6;

using namespace std;

**class node**

{

**private:**

int first=0;

int last=-1;

int count=0;

char a[size];

**public:**

**void Enqueue(char s,int in)**

{

if(!isFull())

{

a[in]=s;

count++;

}

else cout<<"\n\nQUEUE IS FULL\n\n";

}

**int Dequeue()**

{

int ascii;

if(!isEmpty())

{

ascii=int(a[0]);

for(int i=0;i<count;i++)

{

a[i]=a[i+1];

}

count--;

return ascii;

}

else

cout<<"\n\nQUEUE IS EMPTY\n\n";

}

**bool isEmpty()**

{

if(a[0]==NULL)

{

return true;

}

return false;

}

**bool isFull()**

{

if(count==size)

{

return true;

}

return false;

}

**int show()**

{

cout<<endl<<endl;

for(int i=0;i<=count;i++)

{

cout<<a[i];

}

}

};

**int main()**

{

node n;

int opt,in=0;

char s;

cout<<"\t\t\tNAMES";

do

{

cout<<"\n\nWould you like to:\n1. Add a character\n2. Remove a character\n3. Exit\n";

cin>>opt;

if(opt==1)

{

cout<<"\n\nEnter a character : ";

cin>>s;

n.Enqueue(s,in);

in++;

n.show();

}

else if(opt==2)

{

if(!n.isEmpty())

{

cout<<"\nThe ASCII of letter removed is "<<n.Dequeue();

}

else

{

cout<<"\n\nTHE QUEUE IS EMPTY";

}

n.show();

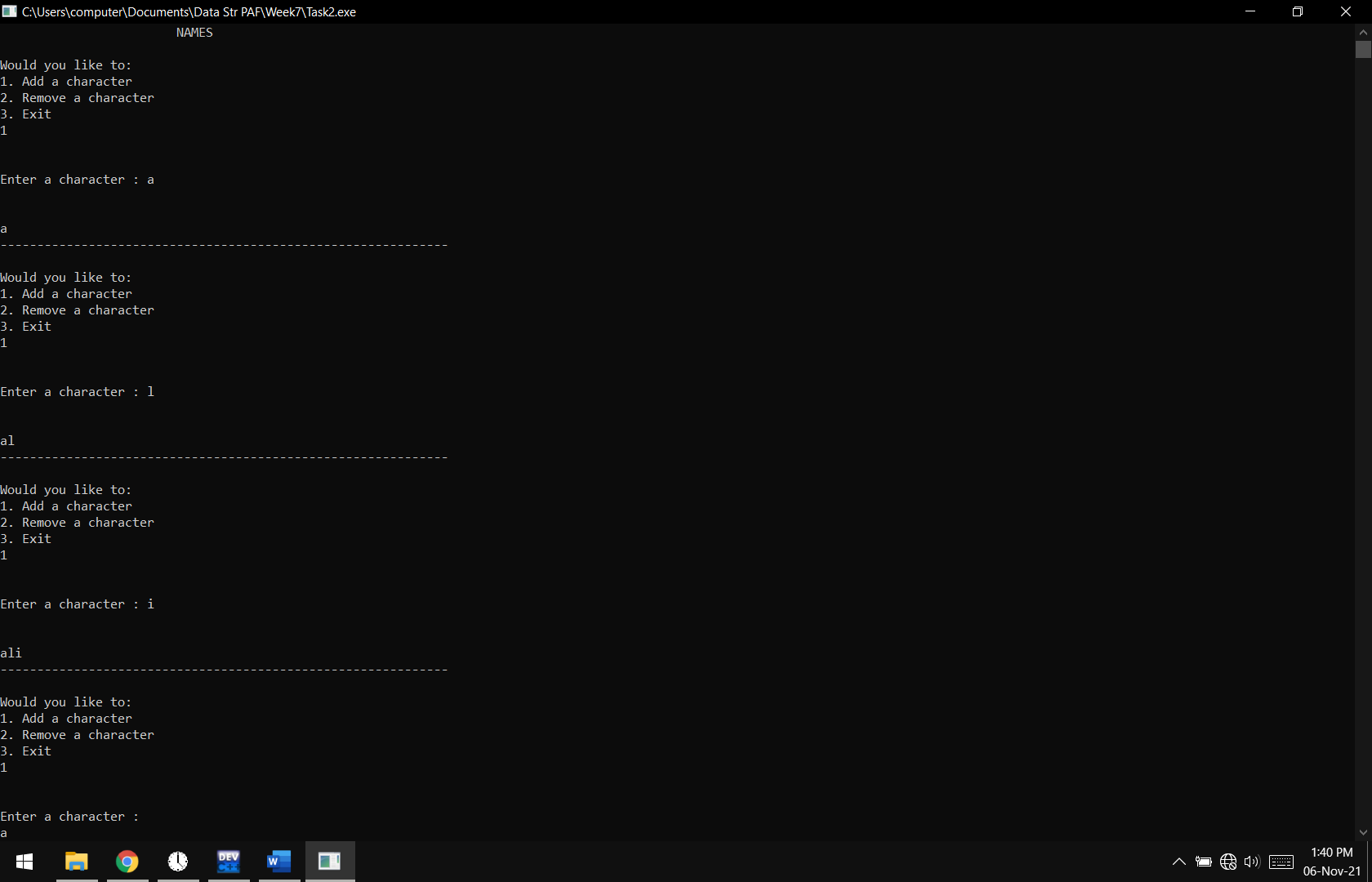
}

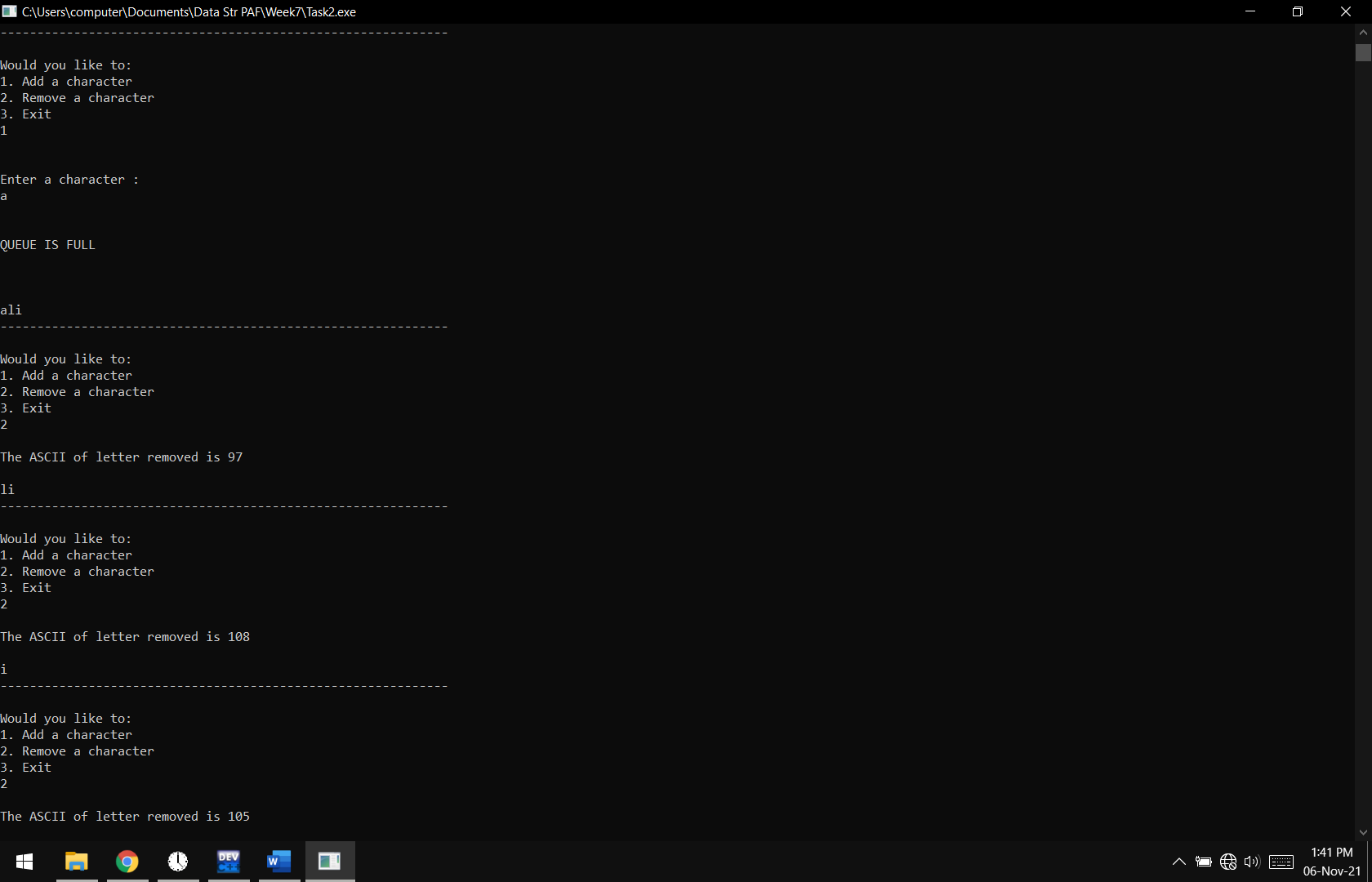
}

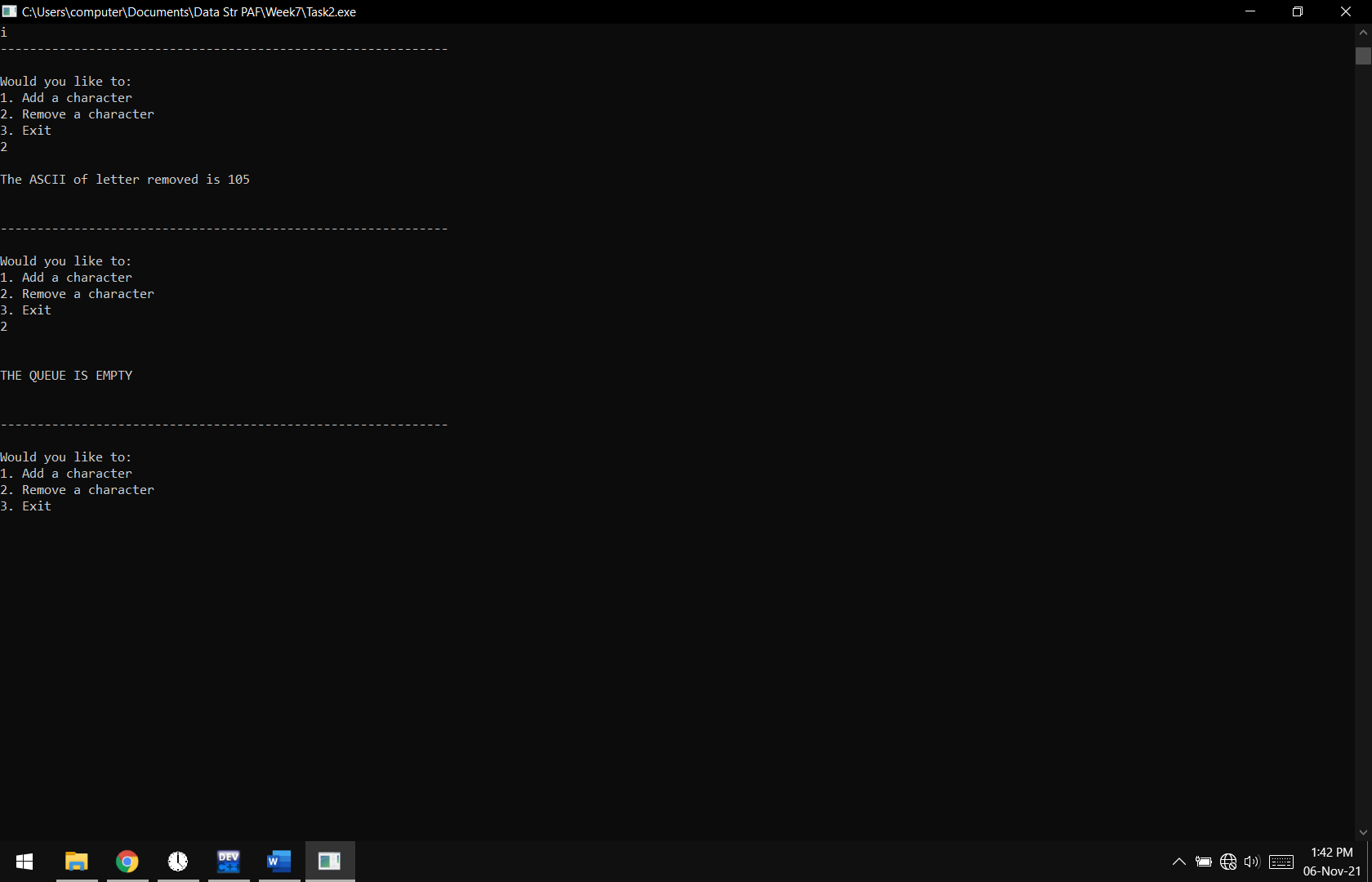
while(opt!=3);

}

**Output:**







**Lab Task 03:** Create a Queue and add values that are being dequeued.

**USING LINKED LIST:**

**Code:**

#include <iostream>

using namespace std;

**class node**

{

**private:**

int data;

node\* next;

node\* head=NULL;

node\* ptr;

int s=0;

**public:**

**void Enqueue(int s)**

{

node\*temp=new node();

temp->data=s;

temp->next=NULL;

if(head==NULL)

{

head=temp;

ptr=head;

}

else

{

ptr->next=temp;

ptr=ptr->next;

}

}

**int Dequeue()**

{

node\*temp=head;

s=s+temp->data;

head=head->next;

delete temp;

return s;

}

**bool isEmpty()**

{

if(head==NULL)

return true;

return false;

}

**int show()**

{

if(head!=NULL)

{

cout<<"\nThe values in the linked list are :\n\n";

ptr=head;

while(ptr->next!=NULL)

{

cout<<"\t"<<ptr->data;

ptr=ptr->next;

}

cout<<"\t"<<ptr->data;

cout<<"\n--------------------------------------------";

}

else return 0;

}

};

**int main()**

{

node n;

int opt;

int s;

cout<<"\t\t\tNUMBERS";

do

{

cout<<"\n\nWould you like to:\n1. Add a number\n2. Remove a number\n3. Exit\n";

cin>>opt;

if(opt==1)

{

cout<<"\n\nEnter a number : ";

cin>>s;

n.Enqueue(s);

n.show();

}

else if(opt==2)

{

if(!n.isEmpty())

{

cout<<"\n->The sum of Dequeues "<<n.Dequeue()<<endl;

}

else

{

cout<<"\n\nTHE QUEUE IS EMPTY";

}

n.show();

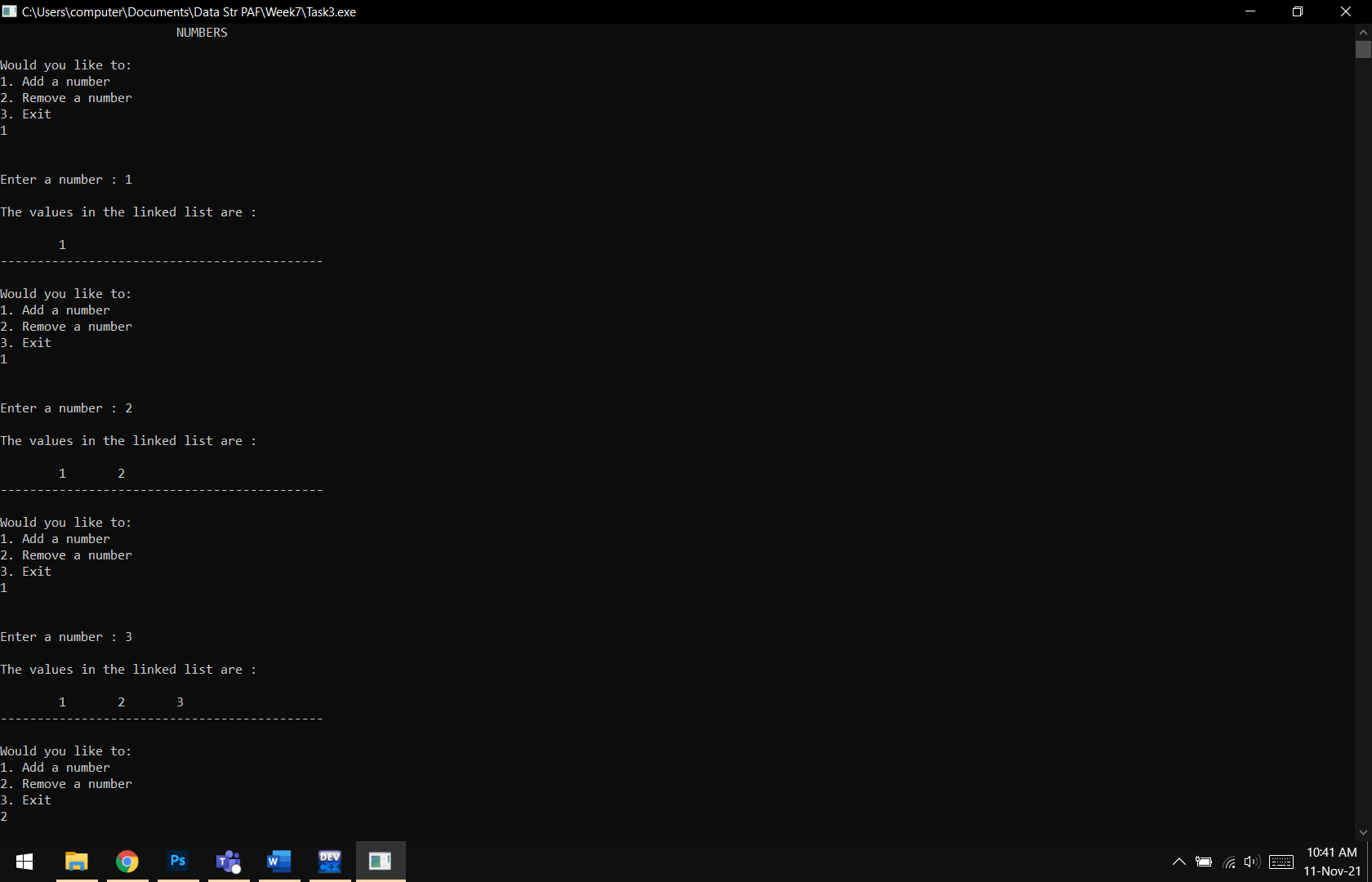
}

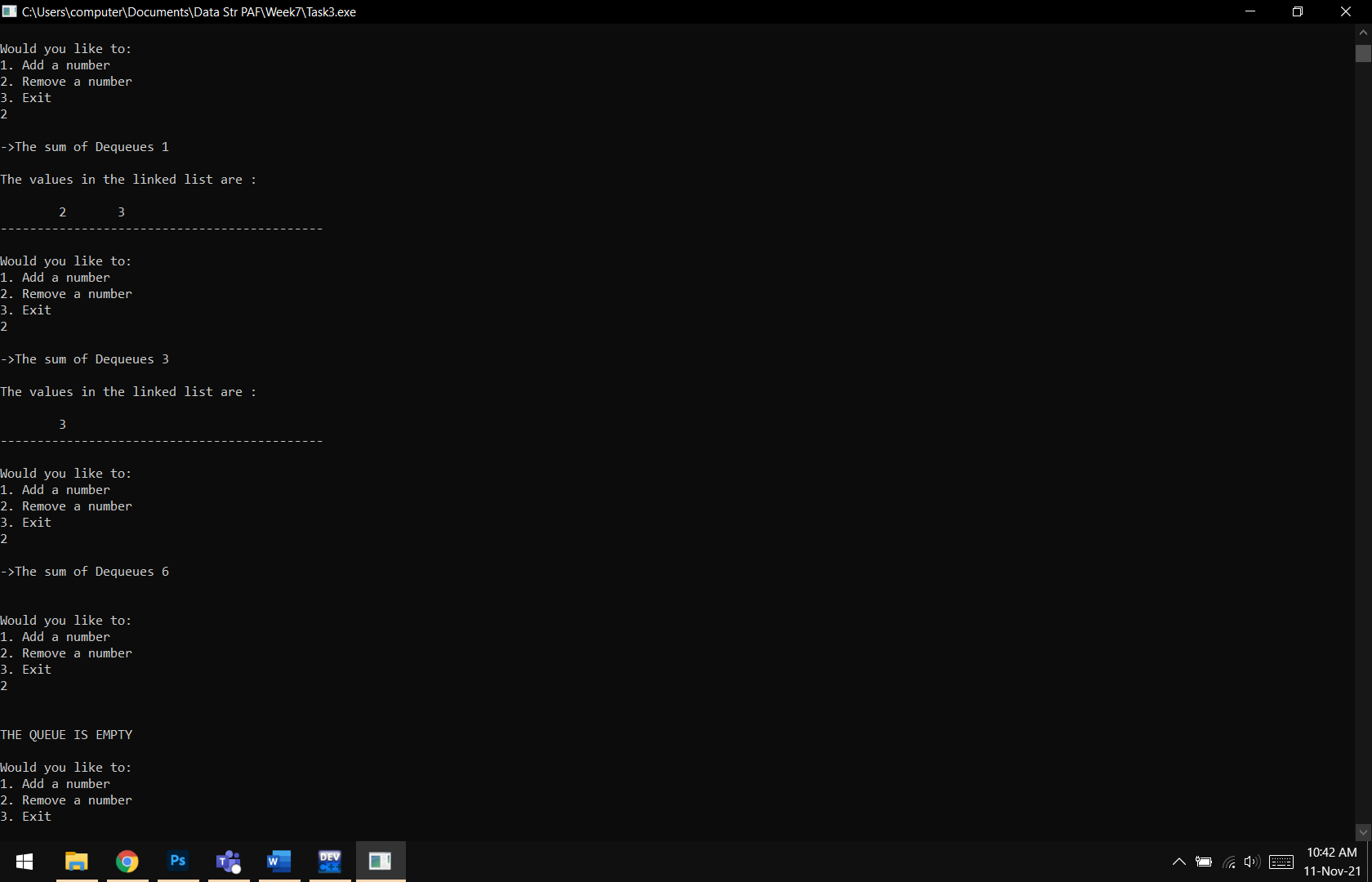
}

while(opt!=3);

}

**Output:**





**Lab Task 04:** Create a Queue and add values that are being dequeued.

**USING ARRAY:**

**Code:**

#include <iostream>

const int size=6;

using namespace std;

**class node**

{

**private:**

int count=0;

int a[size];

int s=0;

**public:**

**void Enqueue(int s,int in)**

{

if(!isFull())

{

a[in]=s;

count++;

}

else cout<<"\n\nQUEUE IS FULL\n\n";

}

int Dequeue()

{

if(!isEmpty())

{

s=s+a[0];

for(int i=0;i<count;i++)

{

a[i]=a[i+1];

}

count--;

return s;

}

else

cout<<"\n\nQUEUE IS EMPTY\n\n";

}

**bool isEmpty()**

{

if(a[0]==NULL)

{

return true;

}

return false;

}

**bool isFull()**

{

if(count==size)

{

return true;

}

return false;

}

int show()

{

cout<<endl<<endl;

for(int i=0;i<count;i++)

{

cout<<a[i];

}

}

};

**int main()**

{

node n;

int opt,in=0;

int s;

cout<<"\t\t\tNUMBERS";

do

{

cout<<"\n\nWould you like to:\n1. Add a number\n2. Remove a number\n3. Exit\n";

cin>>opt;

if(opt==1)

{

cout<<"\n\nEnter a number : ";

cin>>s;

n.Enqueue(s,in);

in++;

n.show();

}

else if(opt==2)

{

if(!n.isEmpty())

{

cout<<"\nThe sum of numbers removed is "<<n.Dequeue();

}

else

{

cout<<"\n\nTHE QUEUE IS EMPTY";

}

n.show();

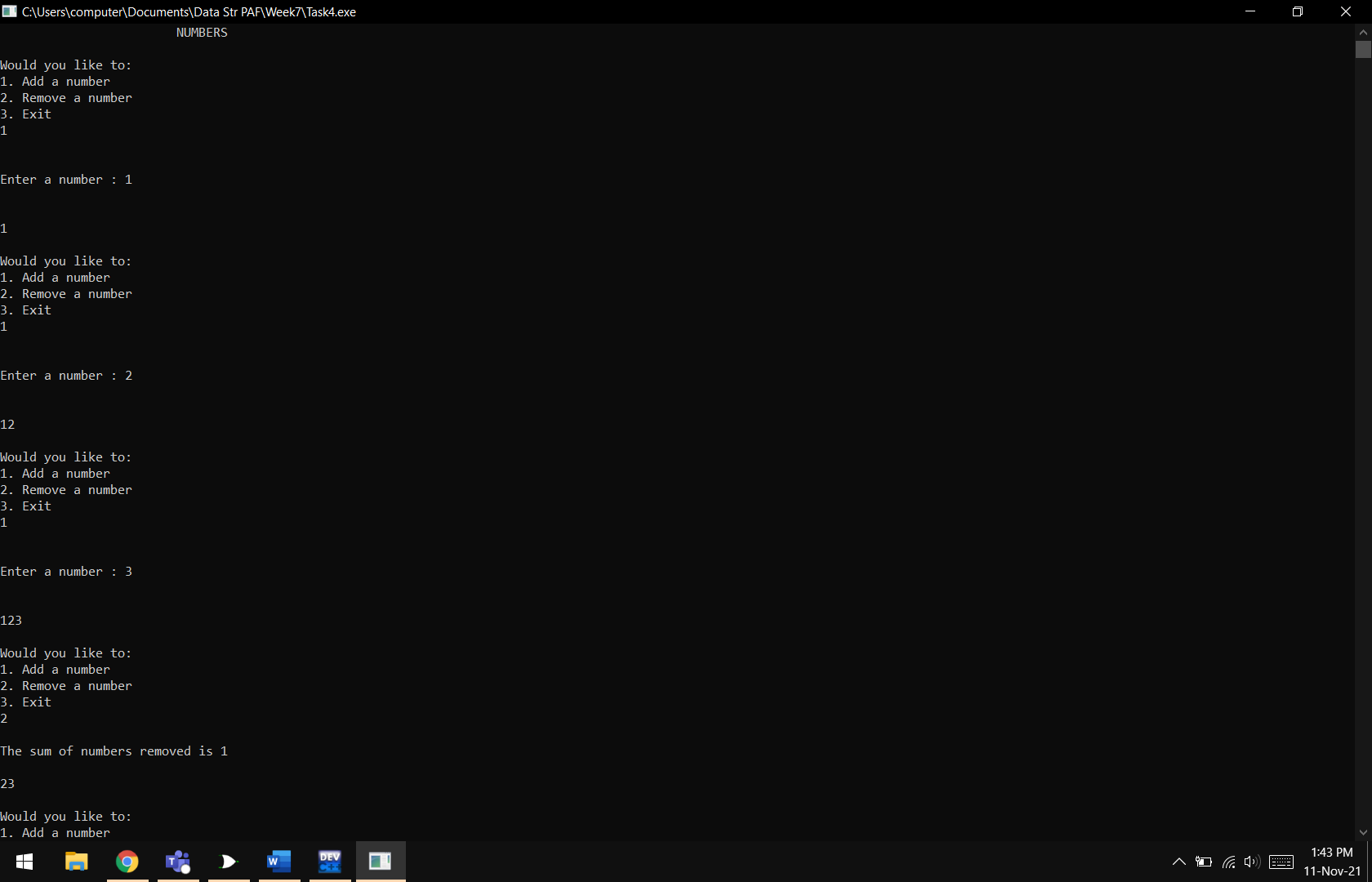
}

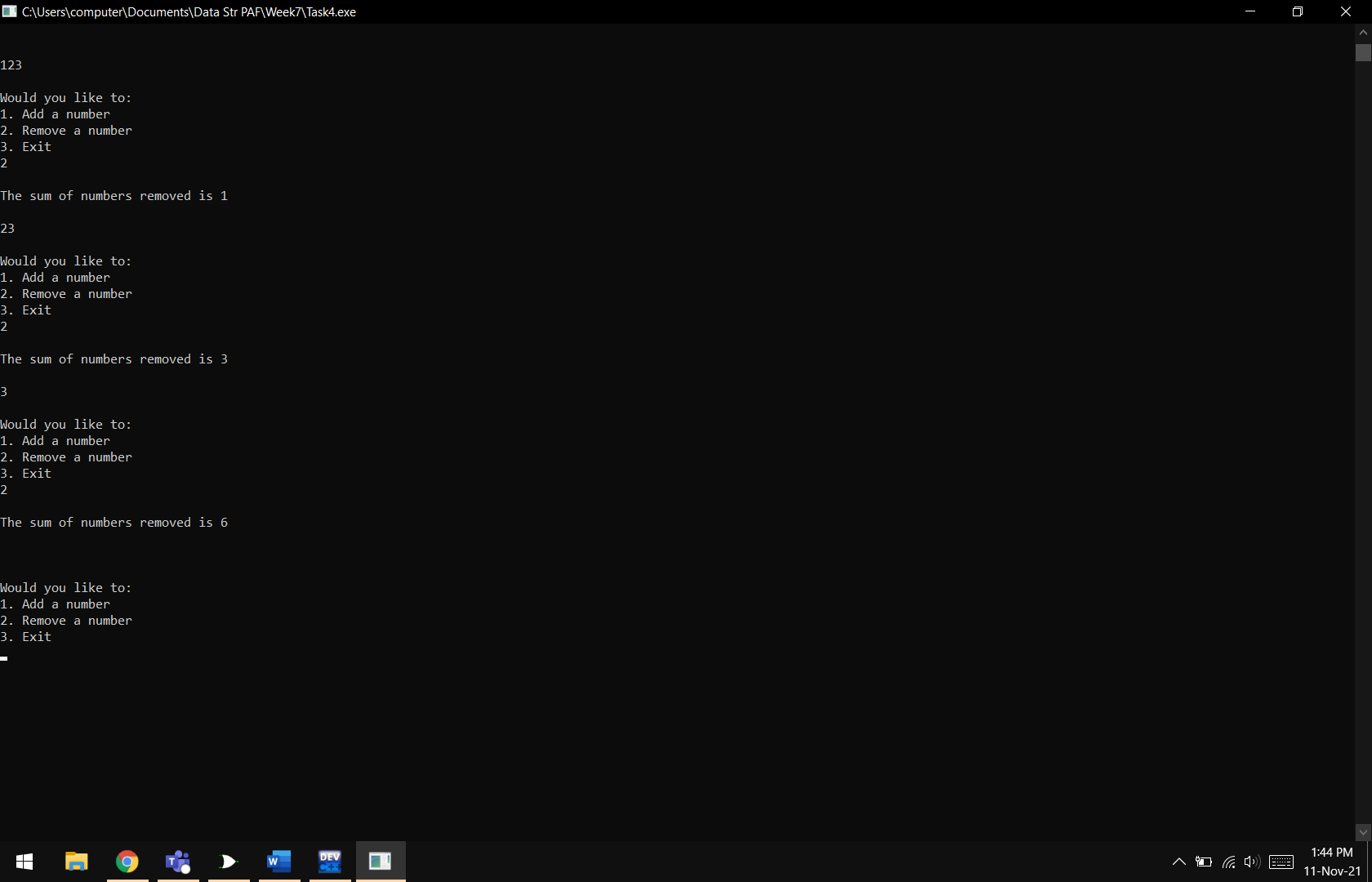
}

while(opt!=3);

}

**Output:**





**Results & Observations:**

In this lab, we have learnt about the basics and implementations of the queue data structure. We can use this data structure through linked lists as well as arrays. The queue is used in computers for data processing and managing. Any data that comes first, should be managed first then move to the next instruction. This is the logic of the queue data structure.