Q: - Write a program to create a linked list.

Input:-

Linked List:-

class LinkedList

{

int length;

Node start;

public LinkedList()

{

start = new Node();

}

public LinkedList(Node obj)

{

start = obj;

}

public bool Underflow()

{

if (start.next == null)

{

return true;

}

else

{

return false;

}

}

/\*Insertion Methods\*/

public bool InsertAtBeg(int val)

{

Node n = new Node(val);

if (!Underflow())

{

n.next = start.next;

start.next = n;

return true;

}

else

{

start.next = n;

return true;

}

}

public bool InsertAtEnd(int val)

{

Node n = new Node(val);

if (!Underflow())

{

Node temp = start.next;

while (temp.next!=null)

{

temp = temp.next;

}

temp.next = n;

return true;

}

else

{

start.next = n;

return true;

}

}

public bool InsetAfter(int oldval,int newval)

{

Node n = new Node(newval);

if (!Underflow())

{

Node temp = start.next;

while (temp.data==newval || temp.next!=null)

{

if (temp.data==oldval)

{

n.next = temp.next;

temp.next = n;

return true;

}

else

{

temp = temp.next;

}

}

if (InsertAtEnd(newval))

{

Console.WriteLine("Value not found");

Console.WriteLine("New value inserted at the end");

return true;

}

}

return false;

}

/\*Remove Methods\*/

public bool RemoveFirst()

{

if (!Underflow())

{

if (start.next.next!=null)

{

start.next = start.next.next;

return true;

}

else

{

start.next = null;

return true;

}

}

else

{

return false;

}

}

public bool RemoveLast()

{

if (!Underflow())

{

if (start.next.next == null)

{

start.next = null;

return true;

}

else

{

Node temp = start.next;

while (temp.next.next != null)

{

temp = temp.next;

}

temp.next = null;

return true;

}

}

return false;

}

public bool RemoveAfter(int val)

{

if (!Underflow())

{

Node temp = start.next;

while (temp.data==val || temp.next!=null)

{

if (temp.data==val)

{

if (temp.next.next!=null)

{

temp.next = temp.next.next;

return true;

}

else

{

temp.next = null;

return true;

}

}

temp = temp.next;

}

}return false;

}

public void Display()

{

if (!Underflow())

{

Node temp = start.next;

Console.WriteLine("Your List ");

while (temp.next!=null)

{

Console.Write(temp.data+" ");

temp = temp.next;

}

Console.Write(temp.data);

Console.WriteLine();

}

else

{

Console.WriteLine("List is Empty");

}

}

Node:-

class Node

{

internal Node next;

internal int data;

public Node()

{

this.next = null;

this.data = 0;

}

public Node(int val)

{

this.data = val;

}

public Node(Node obj)

{

this.next = obj;

}

}

Main:-

LinkedList obj = new LinkedList();

while (true)

{

Console.WriteLine("1-Insert at Beginning\n2-Insert After\n3-Insert at End\n4-Remove First\n5-Remove After\n6-Remove Last\n0-Exit");

Console.WriteLine("What you want to do with LinkList\nEnter 0-6");

int opt = int.Parse(Console.ReadLine());

switch (opt)

{

case 1:

Console.Write("Enter value: ");

int val1=int.Parse(Console.ReadLine());

obj.InsertAtBeg(val1);

obj.Display();

break;

case 2:

Console.Write("Enter Old value : ");

int val2 = int.Parse(Console.ReadLine());

Console.Write("Enter New value : ");

int val00 = int.Parse(Console.ReadLine());

obj.InsetAfter(val2, val00);

obj.Display();

break;

case 3:

Console.Write("Enter value: ");

int val3 = int.Parse(Console.ReadLine());

obj.InsertAtEnd(val3);

obj.Display();

break;

case 4:

obj.RemoveFirst();

obj.Display();

break;

case 5:

Console.Write("Enter Old value: ");

int val5 = int.Parse(Console.ReadLine());

obj.RemoveAfter(val5);

obj.Display();

break;

case 6:

obj.RemoveLast();

obj.Display();

break;

default:

Console.WriteLine("Wrong Input");

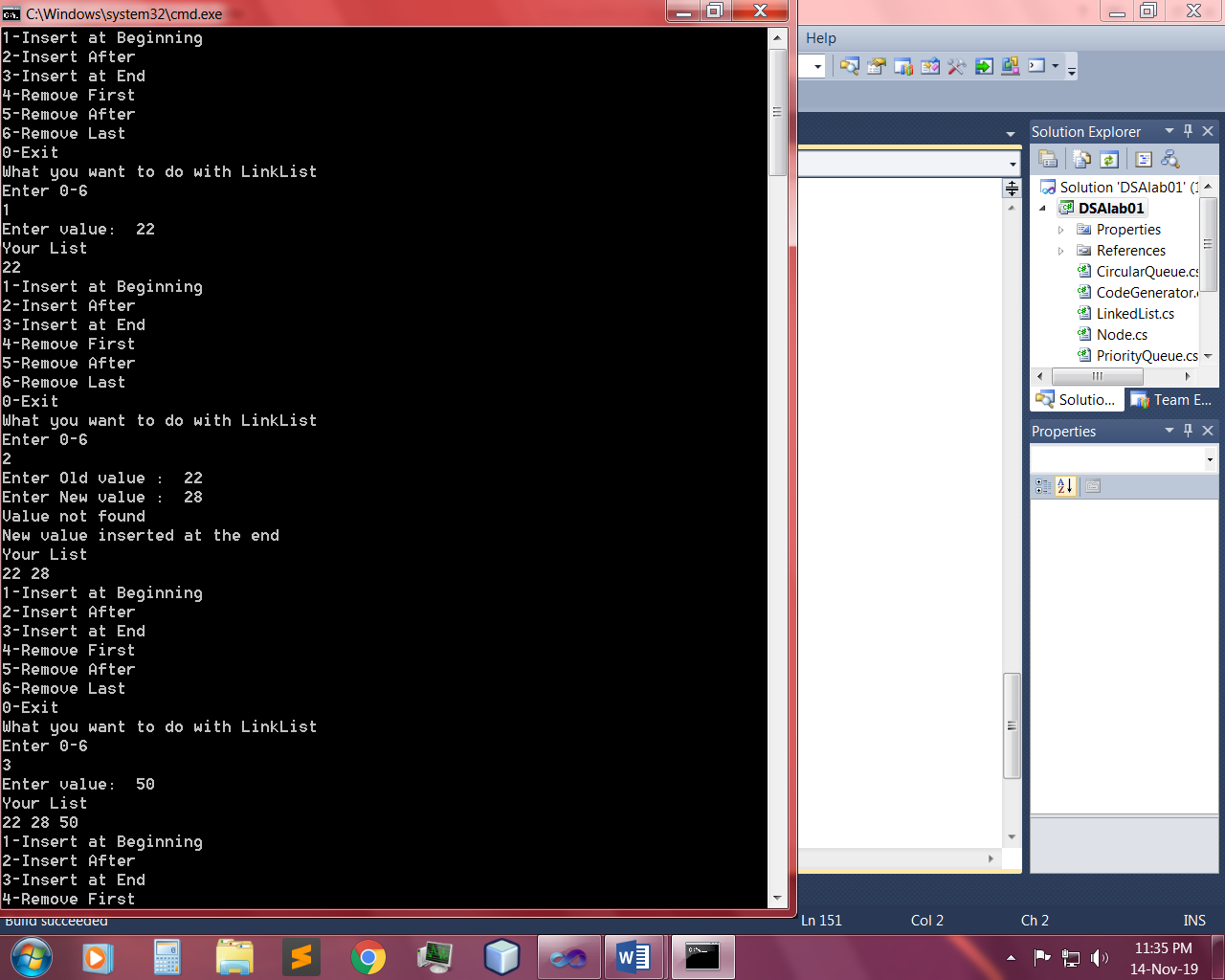
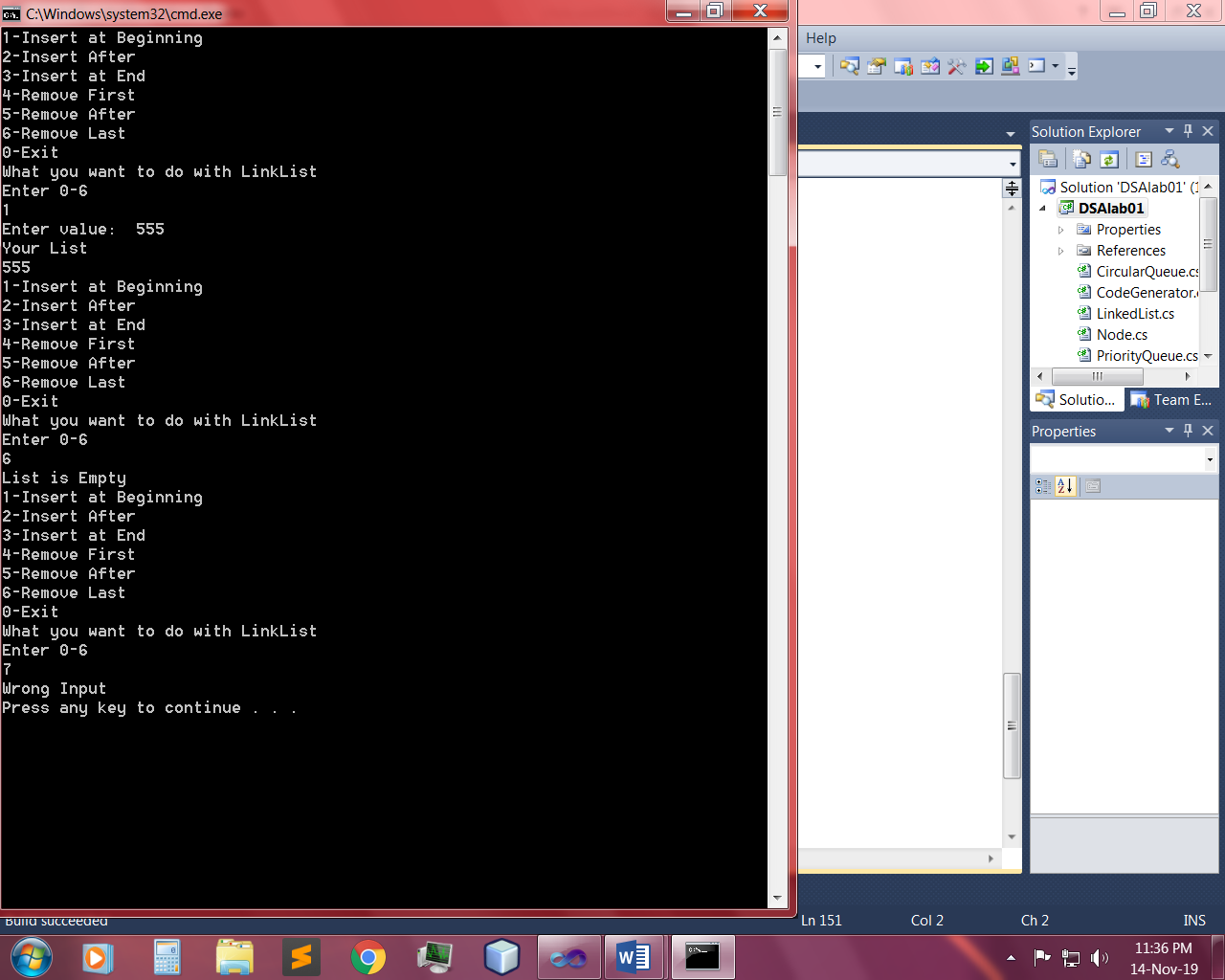
Environment.Exit(0);

break;

}

}

Output:-

Q: - Write a program to implement dynamic stacks by using linked list.

Input:-

Dynamic Stack:-

class DynamicStack

{

LinkedList obj;

public DynamicStack()

{

obj = new LinkedList();

}

public bool Push(int val)

{

return obj.InsertAtEnd(val);

}

public bool Pop()

{

return obj.RemoveLast();

}

public void Peek()

{

Console.WriteLine(obj.Peek());

}

public void Display()

{

obj.Display();

}

}

LinkList:-

public void Display()

{

if (!Underflow())

{

Node temp = start.next;

Console.WriteLine("Your Stack ");

while (temp.next!=null)

{

Console.Write(temp.data+" ");

temp = temp.next;

}

Console.Write(temp.data);

Console.WriteLine();

}

else

{

Console.WriteLine("Stack is Empty");

}

}

public int Peek()

{

if (!Underflow())

{

Node temp = start.next;

if (temp.next!=null)

{

while (temp.next.next != null)

{

temp = temp.next;

}

int val=temp.next.data;

return val;

}

else

{

int val1 = temp.data;

return val1;

}

}

else return - 1;

}

Main:-

DynamicStack obj = new DynamicStack();

while (true)

{

Console.WriteLine("What action you want to perform");

Console.WriteLine("1-Push\n2-Pop\n3-Peek\n0-Exit");

int opt = int.Parse(Console.ReadLine());

if (opt==1)

{

Console.WriteLine("Enter value to push in Stack");

int val=int.Parse(Console.ReadLine());

obj.Push(val);

obj.Display();

}

else if (opt==2)

{

obj.Pop();

obj.Display();

}

else if (opt==3)

{

obj.Peek();

}

else if(opt==0)

{

Console.WriteLine("Successfully Exit.");

Environment.Exit(0);

}

else

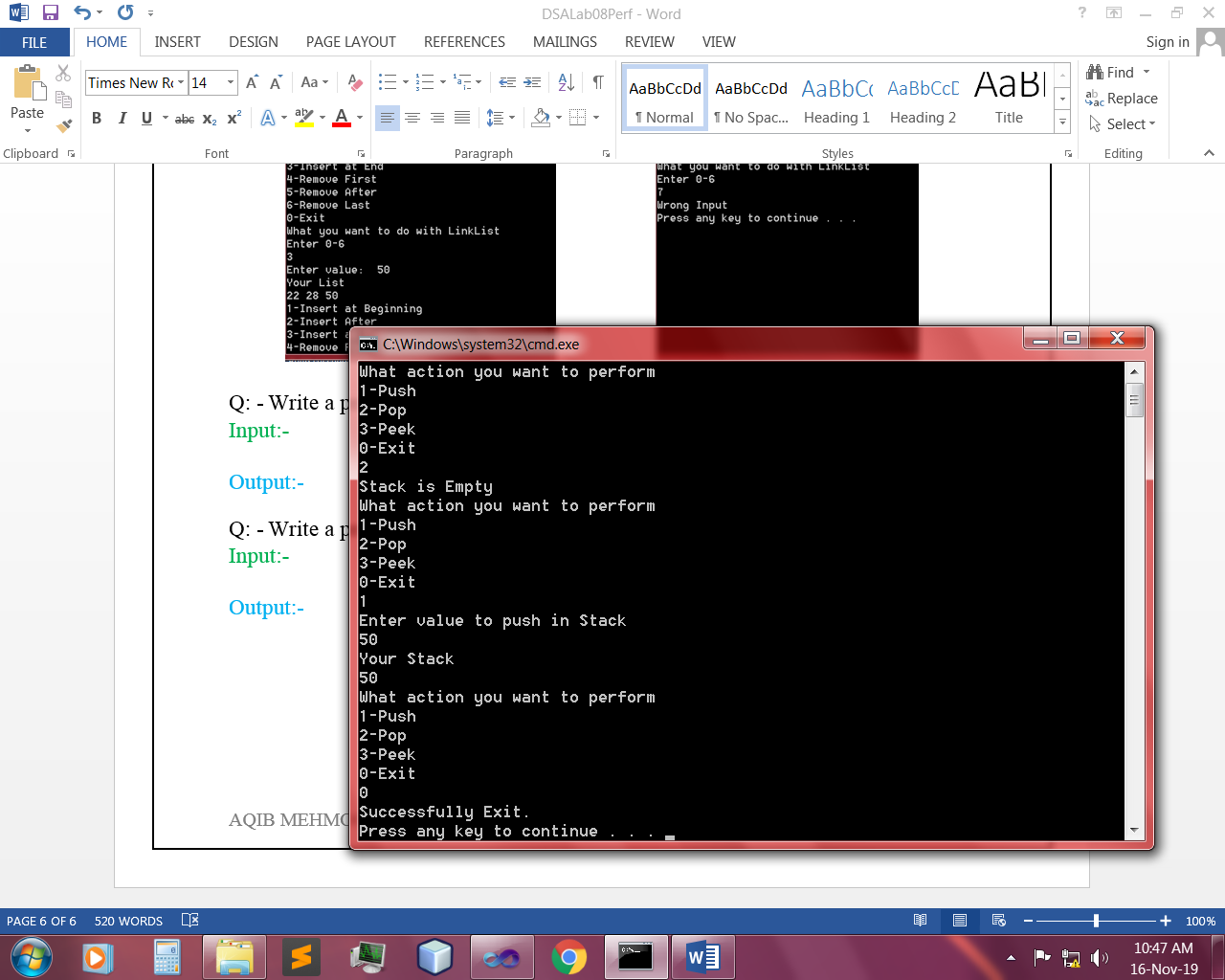
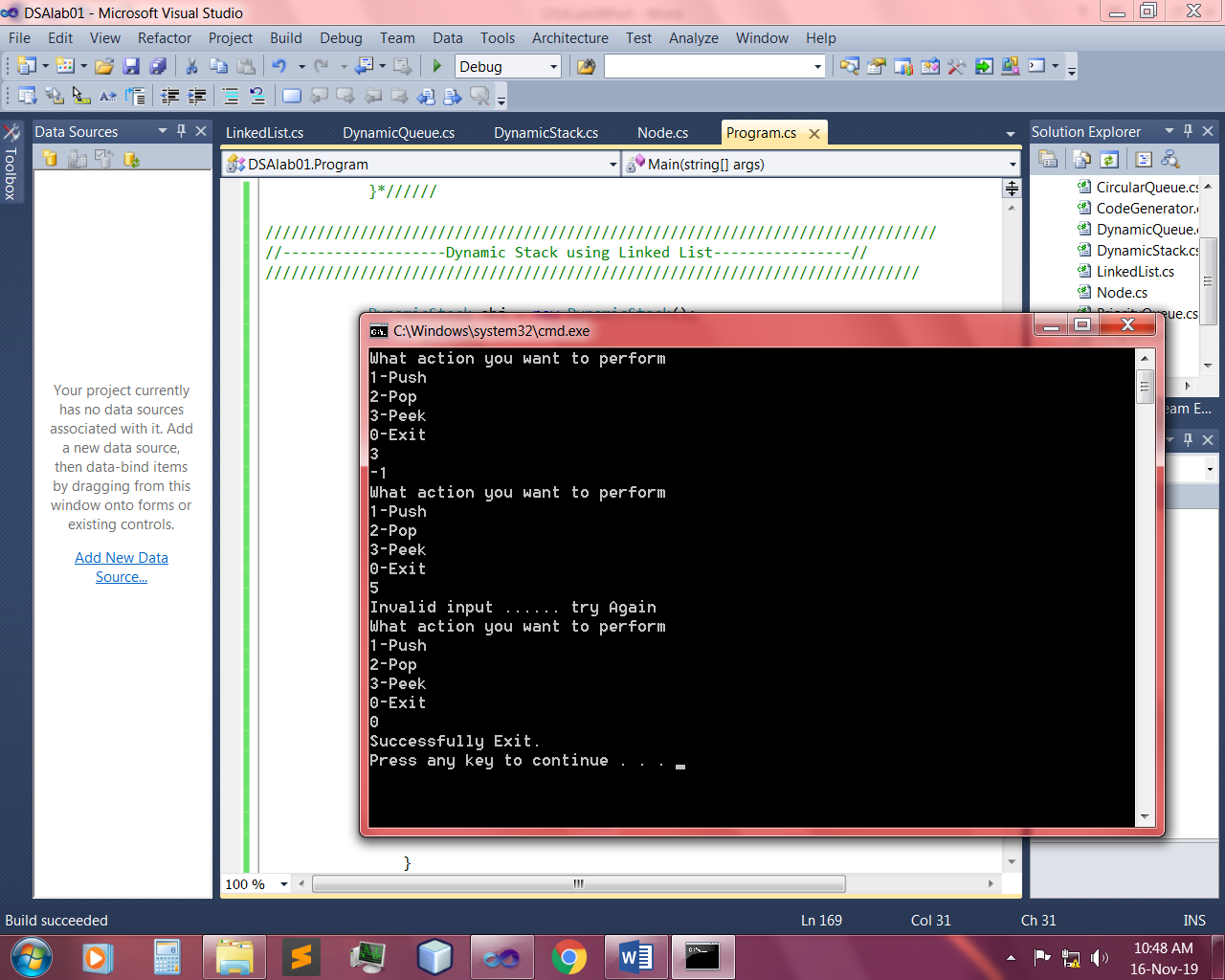
{

Console.WriteLine("Invalid input ...... try Again");

}

}

Output:-

Q: - Write a program to implement dynamic queue by using linked list.

Input:-

Dynamic Queue:-

class DynamicQueue

{

LinkedList obj;

public DynamicQueue()

{

obj = new LinkedList();

}

public bool Enqueue(int val)

{

return obj.InsertAtEnd(val);

}

public bool Dequeue()

{

return obj.RemoveFirst();

}

public void Display()

{

obj.Display();

}

}

Link list:-

public void Display()

{

if (!Underflow())

{

Node temp = start.next;

Console.WriteLine("Your Queue ");

while (temp.next!=null)

{

Console.Write(temp.data+" ");

temp = temp.next;

}

Console.Write(temp.data);

Console.WriteLine();

}

else

{

Console.WriteLine("Queue is Empty");

}

}

Main:-

DynamicQueue obj = new DynamicQueue();

while (true)

{

Console.WriteLine("What action you want to Perform?");

Console.WriteLine("1-Enqueue\n2-Dequeue\n0-Exit");

int opt = int.Parse(Console.ReadLine());

if (opt==1)

{

Console.WriteLine("Enter value");

int val=int.Parse(Console.ReadLine());

obj.Enqueue(val);

obj.Display();

}

else if (opt==2)

{

obj.Dequeue();

obj.Display();

}

else if (opt==0)

{

Console.WriteLine("Successfully Exit");

Environment.Exit(0);

}

else

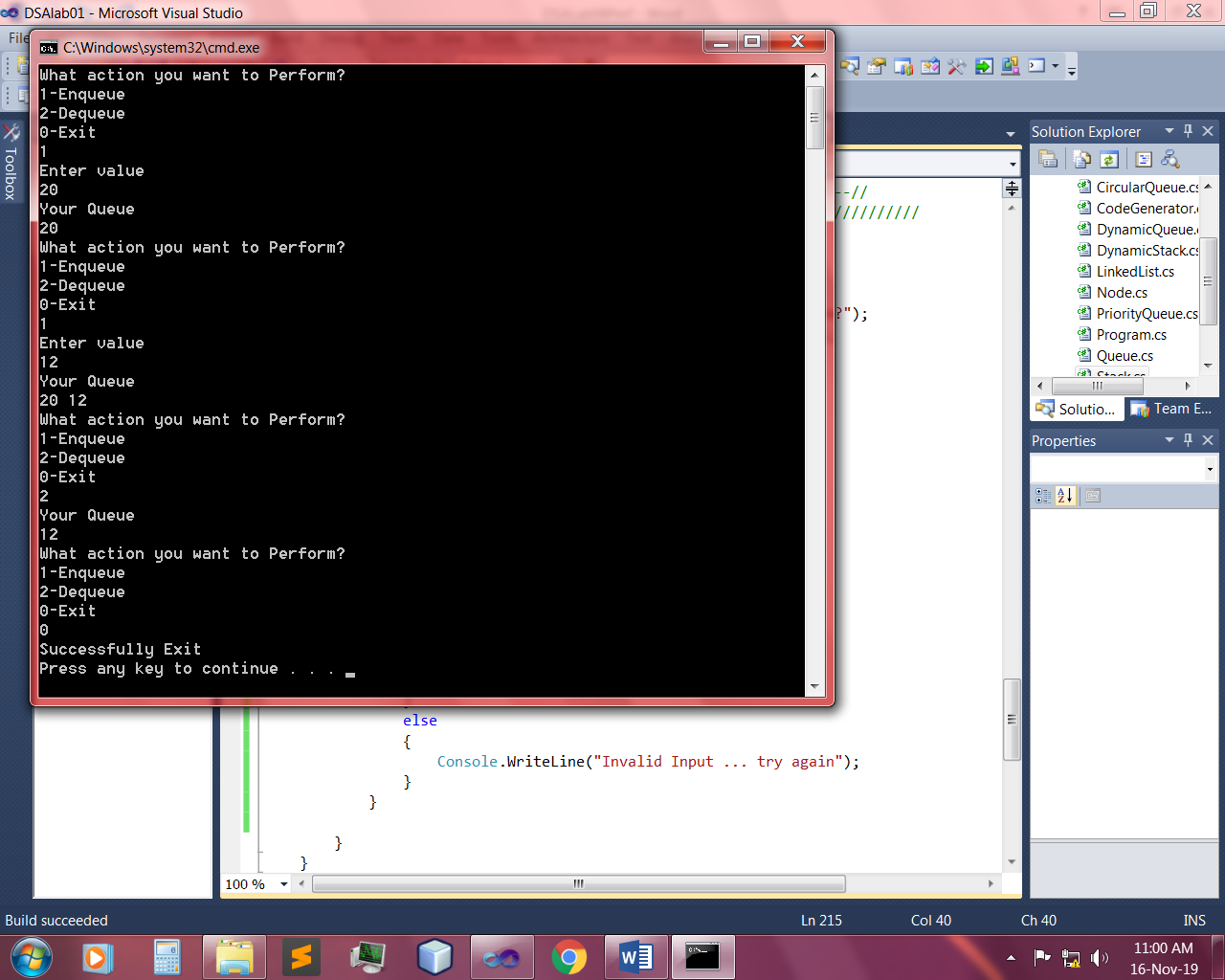
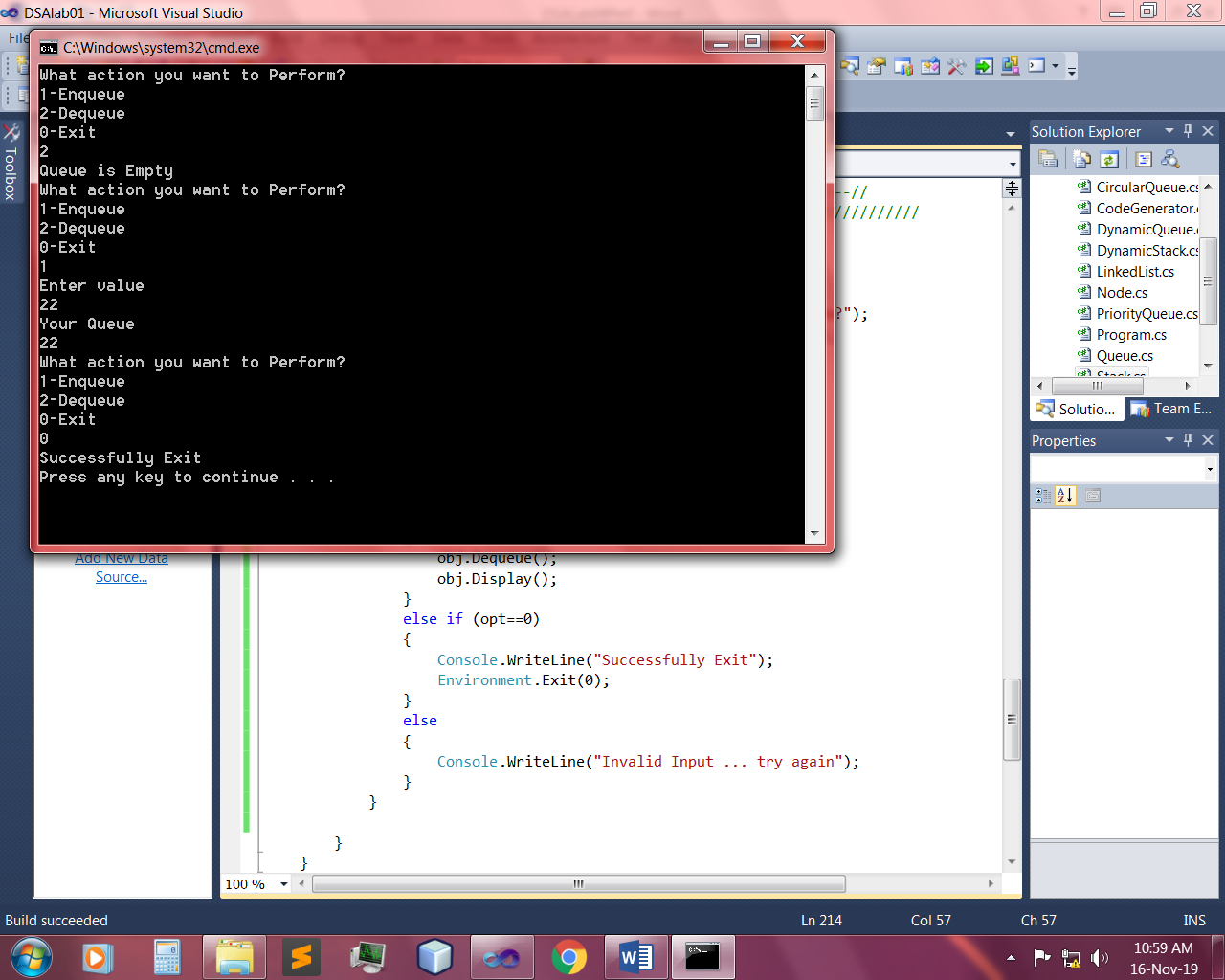
{

Console.WriteLine("Invalid Input ... try again");

}

}

Output:-

~~~~~~\*\*/**THE END**/\*\*~~~~~~