Bahria University,

Karachi Campus

A picture containing text, room

Description automatically generated

LAB EXPERIMENT NO.

\_\_\_\_\_\_

LIST OF TASKS

|  |  |
| --- | --- |
| TASK NO | OBJECTIVE |
| 01 | **Write a program to build your own stack class using linked list. The minimum your stack class should include is using your enrollment no :**  **A Push() method**  **A Pop() method**  **A Peek() method**  **A IsFull() method**  **A IsEmpty() method**  **A Display() method**  **A Count() method** |
|  |  |
|  |  |

Submitted On:

\_\_\_\_\_\_\_\_\_\_\_

(Date: DD/MM/YY)

Task 01: **Write a program to build your own stack class using linked list. The minimum your stack class should include is using your enrollment no :**

1. internal class Node
2. {
3. public string data;
4. public Node next;
5. public Node() {
6. Console.Write("Enter Data : ");
7. this.data = Console.ReadLine();
8. next = null;
9. }
10. }
11. internal class Stack
12. {
13. static int counter;
14. public static Node head;
15. internal void push()
16. {
17. if (head == null)
18. {
19. head = new Node();
20. counter++;
21. return;
22. }
24. Node new\_node = new Node();
25. new\_node.next = null;
27. Node last = head;
28. while (last.next != null)
29. last = last.next;
31. last.next = new\_node;
32. counter++;
33. return;
34. }
36. public void menu(Stack stack) {
37. int inp;
38. Console.WriteLine("/\*------Stacks-------\*/");
39. Console.WriteLine("/\*-------------------\*/\n");
40. Console.Write("Choose One Option...\n" +
41. "1. Push\n" +
42. "2. Pop\n" +
43. "3. Is Empty?\n" +
44. "4. Display\n" +
45. "5. Count\n" +
46. "Enter Option : ");
47. int opt = Convert.ToInt32(Console.ReadLine());
48. switch (opt)
49. {
50. case 1:
51. stack.push(); break;
52. case 2:
53. string last = stack.pop();
54. Console.WriteLine("The Popped Item Is {0}", last);
55. break;
56. case 3:
57. Console.WriteLine("isEmpty? = {0}", isEmpty()); break;
58. case 4:
59. printList(); break;
60. case 5:
61. Console.WriteLine("The Total Item In The List Are: {0}", stack.getCount());
62. break;
63. default:
64. Console.WriteLine("Wrong Input..!!");
65. break;
66. }
67. Console.WriteLine();
68. menu(stack);
69. }
70. internal string pop()
71. {
72. Node last = head, prev = null;
73. if (head == null) {
74. Console.WriteLine("List Is Empty...!!");
75. return null;
76. }
78. while (last.next != null) {
79. prev = last;
80. last = last.next;
81. }
82. prev.next = last.next;
83. counter--;
85. if (last != null) {
86. return last.data;
87. }
88. return null;
89. }
90. internal void peek() {
91. Node last = head;
92. while (last.next != null)
93. last = last.next;
94. Console.WriteLine("The Topmost element Of Stack is : {0}", last.data);
95. }
96. internal void printList()
97. {
98. Node node = head;
99. while (node != null) {
100. Console.WriteLine(node.data + " ");
101. node = node.next;
102. }
103. }
105. internal void printStack()
106. {
107. if (counter <= 0) {
108. Console.WriteLine("Stack Underflow");
109. return;
110. }
111. else {
112. Node node = head;
113. Console.Write("Items In The Stack Are : ");
114. for (int i = counter; i >= 0; i--)
115. {
116. Console.WriteLine(node.data + " ");
117. node = node.next;
118. }
119. Console.WriteLine();
120. }
121. }
122. public int getCount() {
123. return counter;
124. }
125. public bool isEmpty() {
126. return head == null;
127. }
129. static void Main(string[] args)
130. {
131. Stack stack = new Stack();
132. stack.menu(stack);
133. }

Solution:

