

东莞城市学院人工智能学院

School of Artificial Intelligence

《数据结构》

实验报告

| 学号 | 202435710252 |
| --- | --- |
| 姓名 | 钟润柯 |
| 专业、班级 | 2024级计科（专升本）2班 |
| 课程名称 | 《数据结构》 |
| 指导教师 | 李言一 |
| 学期 | 2024—2025第1学期 |
| 实验次序 | 实验三 |
| 上机时间 | 2024-10-23 周三 第5-6节 |
| 上机地点 | 3B416 |
| 分数 |  |

二○二三——二○二四学年 第一学期

实验三 栈的定义和运算

# 实验课时

课内：2课时

# 实验类型

操作性实验

# 实验目的

1.熟练掌握栈的特点。

2.掌握栈的定义和基本操作。

3.熟练掌握顺序栈和链栈的操作及应用。

4.能编程实现顺序栈和链栈的创建、进栈和出栈等。

5.逐步培养解决实际问题的编程能力。

# 实验内容和步骤

## 一、源代码

（1）顺序栈表结构的定义及初始化栈表的源码

定义部分、函数部分和主函数生效部分：

|  |
| --- |
| 1. #include <stdio.h> 2. #include <stdlib.h> 3. #define MAXSIZE 100 4. // 顺序栈的结构定义 5. typedef struct { 6. int data[MAXSIZE]; 7. int top; 8. } Stack; 9. // 顺序站的初始化 10. void InitStack(Stack \*S) { 11. S->top = -1; 12. } 13. // 顺序栈是否为空 14. int StackEmpty(Stack \*S) { 15. return S->top == -1; 16. } 17. void PrintStack(Stack \*S) { 18. while (!StackEmpty(S)) { 19. int x; 20. Pop(S, &x); 21. printf("%d ", x); 22. } 23. printf("\n"); 24. } 25. int main() { 26. Stack S; 27. InitStack(&S); 28. // 任务1 29. for (int i=1;i<=5;i++){ 30. Push(&S, i); 31. } 32. printf("出栈序列"); 33. PrintStack(&S); 34. // 初始化顺序栈 35. InitStack(&S); 36. Push(&S, 3); 37. Push(&S, 2); 38. Push(&S, 1); 39. int s; 40. Pop(&S, &s); 41. printf("出栈：%d\n", s); 42. Pop(&S, &s); 43. printf("出栈：%d\n", s); 44. Push(&S, 4); 45. Push(&S, 5); 46. Push(&S, 6); 47. Push(&S, 7); 48. printf("全部出栈："); 49. PrintStack(&S); 50. } |

（2）顺序栈表插入数据的源码：

|  |
| --- |
| 1. // 顺序栈进栈操作 2. int Push(Stack \*S, int x) { 3. if (S->top == MAXSIZE - 1) { 4. return 0; 5. } 6. S->data[++S->top] = x; 7. return 1; 8. } |

（3）顺序栈表删除数据的源码：

|  |
| --- |
| 1. // 顺序栈出栈操作 2. int Pop(Stack \*S, int \*x) { 3. if (S->top == -1) { 4. return 0; 5. } 6. \*x = S->data[S->top--]; 7. return 1; 8. } |

（4）链栈表结构的定义及初始化链栈表的源码

定义部分、函数部分和主函数生效部分：

|  |
| --- |
| 1. #include <stdio.h> 2. #include <stdlib.h> 3. typedef struct StackNode{ 4. int data; 5. struct StackNode\* next; 6. } StackNode, \*LinkStack; 7. // 创建链栈 8. LinkStack CreateStack() { 9. LinkStack s = (LinkStack) malloc (sizeof(StackNode)); 10. s->next = NULL; 11. return s; 12. } 13. // 链栈为空 14. int StackEmpty(LinkStack s) { 15. return s->next == NULL; 16. } 17. void PrintStack(LinkStack s) { 18. while (!StackEmpty(s)) { 19. int x; 20. Pop(s, &x); 21. printf("%d ", x); 22. } 23. printf("\n"); 24. } 25. int main() { 26. LinkStack s = CreateStack(); 27. // 1. 28. for (int i=1;i<=5;i++){ 29. Push(s, i); 30. } 31. printf("出栈序列："); 32. PrintStack(s); 33. Push(s, 3); 34. Push(s, 2); 35. Push(s, 1); 36. int x; 37. Pop(s, &x); 38. printf("出栈：%d\n", x); 39. Pop(s, &x); 40. printf("出栈：%d\n", x); 41. Push(s, 4); 42. Push(s, 5); 43. Push(s, 6); 44. Push(s, 7); 45. printf("全部出栈："); 46. PrintStack(s); 47. return 0; 48. } |

（5）链栈表插入数据的源码：

|  |
| --- |
| 1. int Push(LinkStack s, int x) { 2. StackNode\* node = (StackNode\*) malloc (sizeof(StackNode)); 3. if (node == NULL) { 4. return 0; 5. } 6. node->data = x; 7. node->next = s->next; 8. s->next = node; 9. return 1; 10. } |

（6）链栈表删除数据的源码：

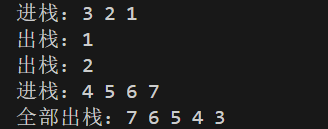
|  |
| --- |
| 1. int Pop(LinkStack s, int\* x) { 2. if (StackEmpty(s)) { 3. return 0; 4. } 5. StackNode\* node = s->next; 6. \*x = node->data; 7. s->next = node->next; 8. free(node); 9. return 1; 10. } |

## 二、程序运行结果截图

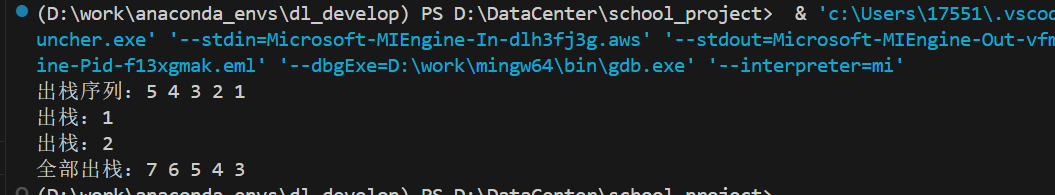
（1）顺序栈表任务1截图：



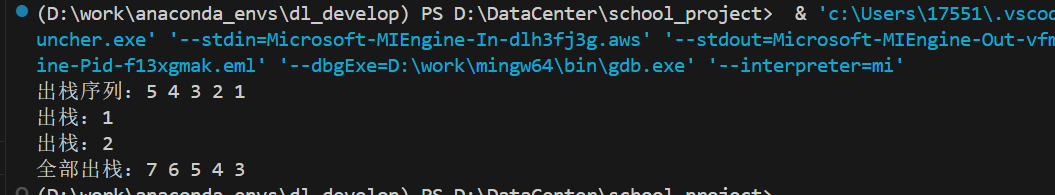
（2）顺序栈表任务2截图：



（3）链栈任务1截图：



（4）链栈任务2截图：



## 三、实验总结

通过本次实验，我深入理解了栈的基本概念和操作，掌握了顺序栈和链栈的实现方法。实验过程中，我编写了顺序栈和链栈的初始化、进栈、出栈等基本操作函数，并成功完成了指定的任务。

在任务1中，通过顺序栈实现了数据的进栈和出栈操作，验证了栈的“后进先出”特性。任务2中，利用链栈实现了相同的功能，进一步巩固了对栈的链式存储结构的理解。

调试过程中，遇到了指针操作和内存管理的问题，通过仔细检查和调试，最终解决了这些问题，提升了编程和调试能力。

此次实验不仅加深了我对栈数据结构的理解，也锻炼了我在实际问题中应用数据结构的能力，为后续学习打下了坚实基础。