

班级\_\_\_\_\_

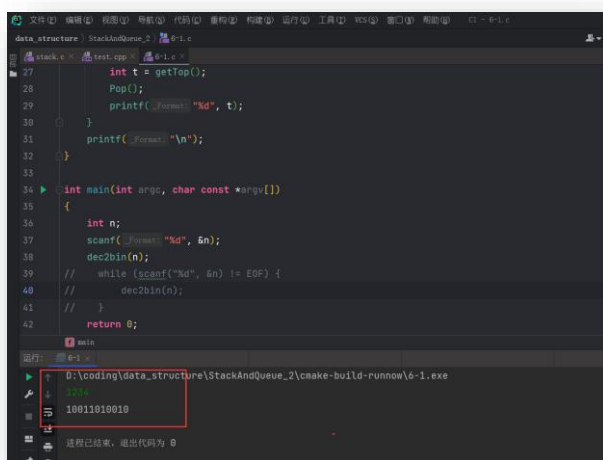
## 实验题目:

### 一、概述

栈和队列的应用关键算法实现及性能分析,能够运用高级程序设计技术实现栈和队列的应用及其关键算法,理解栈和队列的工作原理

### 二、实验过程

#### 1. 调试分析



```
data_structure\StackAndQueue_2\6-1.c
27     int t = getTop();
28     Pop();
29     printf(_Format: "%d", t);
30 }
31 printf(_Format: "\n");
32 }
33
34 int main(int argc, char const *argv[])
35 {
36     int n;
37     scanf(_Format: "%d", &n);
38     dec2bin(n);
39     // while (scanf("%d", &n) != EOF) {
40     //     dec2bin(n);
41     // }
42     return 0;
43 }

main
运行 运行 E-1
B:\coding\data_structure\StackAndQueue_2\cnae-build-runnow\6-1.exe
10011010010
进程已结束, 退出代码为 0
```

#### 2. 测试过程

输入十进制数, 期望输出二进制数

### 三、评价分析

#### 1. 实验结果分析

程序正常运行, 输出对应数字的二进制数

#### 2. 算法性能评价

程序时间复杂度为  $O(n)$

```
#include "stdio.h"
#define MaxSize 100
int top;
int mystack[MaxSize];
typedef enum{false, true} bool;
bool isEmpty();
void Push(int x);
int getTop();
void Pop();
```

```
void dec2bin(int x) {
    top = -1;
    while (x) {
        Push(x % 2);
        x /= 2;
    }
    while (!isEmpty()) {
        int t = getTop();
        Pop();
        printf("%d", t);
    }
    printf("\n");
}

int main()
{
    int n;
    scanf("%d", &n);
    dec2bin(n);
    return 0;
}

bool isEmpty(){
    if(top == -1){
        return true;
    }
    return false;
}

void Push(int num){
    top += 1;
    mystack[top] = num;
}

int getTop(){
    return mystack[top];
}

void Pop(){
    top += -1;
}
```

#### 四、总结与体会

队列：基于地址指针进行遍历，而且可以从头部或者尾部进行遍历，但不能同时遍历，无需开辟空间，因为在遍历的过程中不影响数据结构，所以遍历速度要快

栈：只能从顶部取数据，也就是说最先进入栈底的，需要遍历整个栈才能取出来，遍历数据时需要微数据开辟临时空间，保持数据在遍历前的一致性