递归

2022 年 8 月 16 日

```c
#include <stdio.h>
void print_syracuse_sequence_from(int n){
    printf("%d\t", n);
    if(n == 1) return;
    if(n % 2 == 0)
        print_syracuse_sequence_from(n / 2);
    else
        print_syracuse_sequence_from(3 * n + 1);
}
int main(){
    int n;
    printf("Input a pos for a syracuse sequence:\n");
    scanf("%d", &n);
    print_syracuse_sequence_from(n);
    return 0;
}
```

1. 输入两个正整数，用递归法求最大公约数。

```c
#include <stdio.h>

int gcd(int m, int n)
// 递归求m和n的最大公约数
{
    if(!n) // 上一次得到余数0
    {
        return m;
    }
    return gcd(n, m % n); // 辗转规约
}

int main()
{
    int s, t;
    printf("Input two pos int for greatest common divisors:\n");
    scanf("%d%d", &s, &t);
    int m, n;
    s > t ? (m = s, n = t) : (m = t, n = s); // 保证m>=n
    int d = gcd(m, n);
    printf("The gcd is %d.\n", d);
    return 0;
}
```

2. 求一个输入正整数各个数位上的数字之和。

```c
#include <stdio.h>

int digit_sum(int m)
// 递归求正整数各个数位上数字的和
{
    if(!m) // 上一次得到余数0
    {
        return 0;
    }
    return digit_sum(m / 10) + m % 10;
}

int main()
{
    int n;
    printf("Input a pos int digit sum computation:\n");
    scanf("%d", &n);
    int s = digit_sum(n);
    printf("The digit sum of %d is %d.\n", n, s);
    return 0;
}
```

3. 输入一个字符串和一个字符，查找该字符在该字符串中第一次出现的位置并输出。如果该字符不出现，则输出不出现。

```c
#include <stdio.h>

char* search_for(char* s, char ch)
// 递归
{
    if(!(*s)) return NULL;
    if(*s == ch) return s;
    return search_for(++s, ch);
}

int main()
{
    char str[1024];
    printf("Input a string to be searched:\n");
    gets(str);
    printf("Input a char to be located:\n");
    char ch = getchar();

    char *p = search_for(str, ch);
    if(p)
    {
        printf("1st occurence of '%c' found at %p, at index %d\n", ch, p, p - str);
    }
    else
    {
        printf("'%c' not found\n", ch);
    }

    return 0;
}
```

4. Syracuse（也称为“Collatz”或“Hailstone”）序列的生成从一个自然数开始，重复应用以下函数，直到达到 1：

$$syr(x) = \begin{cases} 3x+1, & \text{当}x\text{为奇数时;} \\ x \div 2, & \text{当}x\text{为偶数时。} \end{cases}$$

例如，从 5 开始的 Syracuse 序列是 5,16,8,4,2,1。数学中有一个悬而未决的问题：对于每个可能的起始值，该序列是否总会到达 1。编程从用户获取起始值，然后打印该起始值的 Syracuse 序列。

```c
#include <stdio.h>

void print_syracuse_sequence_from(int n)
{
    printf("%d\t", n);
    if(n == 1)
    {
        return;
    }
    if(n % 2 == 0)
    {
        print_syracuse_sequence_from(n / 2);
    }
    else
    {
        print_syracuse_sequence_from(3 * n + 1);
    }
}

int main()
{
    int n;
    printf("Input a pos for a syracuse sequence:\n");
    scanf("%d", &n);
    print_syracuse_sequence_from(n);
    return 0;
}
```

5. 请求用户输入两个字符串，然后按字典序比较两个字符串的先后。

```c
#include <stdio.h>
```

```

int string_compare(char *s, char *t)
// 1 means t comes before
// -1 means s comes before
// 0 means equal
{
    if(!(*s) && !(*t)) return 0;
    if(!(*s)) return -1;
    if(!(*t)) return 1;
    if(*s > *t) return 1;
    if(*s < *t) return -1;
    return string_compare(++s, ++t);
}

int main()
{
    char s[1024], t[1024];
    printf("Input two strings for comparison:\n");
    gets(s);
    gets(t);
    int res = string_compare(s, t);
    if(res == 1)
    {
        printf("%s comes before %s\n", t, s);
    }
    else if(res == -1)
    {
        printf("%s comes before %s\n", s, t);
    }
    else
    {
        printf("equal\n");
    }
    return 0;
}
```

6. 在数组中查找最大的元素。

```
#include <stdio.h>
```

```c
#include <limits.h>

int maximum(int *p, int n, int maxi)
// search a list of n element from p
// and return the biggest one among maxi and those n elements
{
    if(n == 0) return maxi;
    if(maxi > *p)
    {
        return maximum(++p, n - 1, maxi);
    }
    else
    {
        return maximum(++p, n - 1, *p);
    }
}

int main()
{
    // int a[] = {1, 3, 7, 2, 4, 3, 2, 5};
    int a[] = {12, 43, 23, 87, 55, 24, 69, 77, 95, 34, 27};
    int m = maximum(a, sizeof(a) / sizeof(int), INT_MIN);
    printf("The greatest: %d\n", m);
    return 0;
}
```