知识点总结

·在C语言中，递归调用是指一个函数在其函数体内部调用自身的过程，它可以将复杂问题分解为规模更小的相似子问题来解决。

递归函数是指通过调用自身来解决问题的函数。

·有效递归需满足三个要素：

1、有明确的终止条件

2、问题的规模逐渐变小

3、子问题与原问题是同类问题

·递归函数包含两个关键部分：基本情况和递归情况。

基本情况：是递归的终止条件，决定递归何时结束的判断依据，避免函数无限递归下去，进而引发程序错误。当满足基本情况时，函数将直接返回一个结果，而不再进行递归调用。

递归情况：在不满足基本情况时，函数会调用自身来解决一个规模更小的同类问题。

·例：计算阶乘，n的阶乘定义为所有小于及等于n的正整数的积，即n! = n * (n-1) * (n-2) * ... * 1，且规定0! = 1。

```
int factorial(int n) {
    // 基本情况
    if (n == 0) {
        return 1;
    }
    // 递归情况
    return n * factorial(n - 1);
}
```

在这个例子中，n == 0是基本情况，当 n 满足此条件时，函数不再递归调用自身，而是直接返回结果。

`return n * factorial(n - 1);` 是递归情况，将求 n 的阶乘分解为求n 乘以 n - 1 的阶乘，通过不断递归调用，最终在满足基本情况时返回结果。

·在上述代码中，程序的执行流程如下：

1、main函数开始执行，定义一个int类型num，并赋初始值为3，接着调用factorial，传入的num作为参数。

2、factorial开始执行，形参n = 3，不满足基本情况，所以执行return n * factorial(n - 1);，也就是执行 return 3 * factorial(2);，此时 factorial(3) 的计算暂停，等待 factorial(2) 的结果。

3、factorial(2) 同样不满足基本情况，所以执行 return 2 * factorial(1);，此时，factorial(2) 的计算暂停，等待 factorial(1) 的结果。

·递归调用的执行过程：

以计算阶乘的递归函数为例：

```c
#include <stdio.h>
int factorial(int n) {
    if (n == 0) {
        return 1;
    }
    return n * factorial(n - 1);
}
int main() {
    int num = 3;
    int result = factorial(num);
    printf("%d的阶乘是%d\n", num, result);
    return 0;
}
```

4、factorial(1) 也不满足基本情况，所以执行 return 1 * factorial(0);，此时，factorial(1) 的计算暂停，等待 factorial(0) 的结果。

5、factorial(0)满足终止条件，返回1。

6、factorial(1) 获得 factorial(0) 的结果1，计算 1 * 1 并返回1。

7、factorial(2) 获得 factorial(1) 的结果1，计算 2 * 1 并返回2。

·递归调用的执行过程：

以计算阶乘的递归函数为例：

```c
#include <stdio.h>
int factorial(int n) {
    if (n == 0) {
        return 1;
    }
    return n * factorial(n - 1);
}
int main() {
    int num = 3;
    int result = factorial(num);
    printf("%d的阶乘是%d\n", num, result);
    return 0;
}
```

8、factorial(3) 获得 factorial(2) 的结果2，计算 3 * 2 并返回6，程序回到main里调用factorial的位置。

9、继续执行main函数中剩余的代码，把factorial函数返回的6赋值给result，然后通过printf打印出结果。

10、main函数执行完毕，程序结束。

·递归调用的执行过程：

以计算阶乘的递归函数为例：

```c
#include <stdio.h>
int factorial(int n) {
    if (n == 0) {
        return 1;
    }
    return n * factorial(n - 1);
}
int main() {
    int num = 3;
    int result = factorial(num);
    printf("%d的阶乘是%d\n", num, result);
    return 0;
}
```

注意：

1、要有明确的终止条件，否则函数会无限递归，进而导致程序崩溃。

2、在递归调用过程中，要确保传递给下一层递归的参数能够使问题规模逐渐缩小，最终满足终止条件。例如在计算阶乘的例子中，每次递归调用 factorial 函数时，n都减1，使得问题规模逐渐减小。

题目1、有以下程序:

```
#include <stdio.h>
int fun(int n)
{
    if (n) return fun(n - 1) + n;
    else return 0;
}
main()
{
    printf("%d\n", fun(3));
}
```

程序的运行结果是（ C ）

A. 4

B. 5

C. 6

D. 7

题目2、以下程序:

```
#include <stdio.h>
void fun(int x)
{
    if (x / 2 > 1) fun(x / 2);
    printf("%d", x);
}
main()
{
    fun(7);
    printf("\n");
}
```

程序运行后的结果是（ D ）

A. 137

B. 731

C. 73

D. 37

题目3、有以下程序:

```
#include <stdio.h>
int fun(int a, int b)
{
    if (b==0) return a;
    else return(fun(--a, --b));
}
main()
{
    printf("%d\n", fun(4, 2));
}
```

程序运行的结果是（B）

A. 1

B. 2

C. 3

D. 4

题目4、有如下程序:

```
#include <stdio.h>
void get_put()
{
    char ch;
    ch = getchar();
    if(ch != '\n') get_put();
    putchar(ch);
}
main()
{
    get_put();
    printf("\n");
}
```

程序运行时，输入1234<回车>，则输出结果是（ B ）

A. 1234

B. 4321

C. 4444

D. 1111

stack

题目5、下列给定程序中，函数fun的功能是：按以下递归公式求函数的值。例如，当给n输入5时，函数值为18；当给n输入3时，函数值为14。请改正程序中的错误，使它能得出正确的结果。

注意：不要改动main函数，不得增行或删行，也不得更改程序的结构！

$$fun(n)=\begin{cases}10 & (n=1)\\ fun(n-1)+2 & (n>1)\end{cases}$$

```
#include <stdio.h>
/**********found**********/
fun(n)    int fun(int n)
{
    int c;
    /**********found**********/
    if(n = 1)   if(n == 1)
        c = 10;
    else
        c = fun(n - 1) + 2;
    return c;
}
int main()
{
    int n;
    printf("Enter n:\n");
    scanf("%d", &n);
    printf("The result is:%d\n\n", fun(n));
}
```