**6.函数指针数组**

用途--->转移表

int add(int x, int y);
int add2(int x, int y);
int add3(int x, int y);
int add4(int x, int y);
int main()
{
//指针数组
int\* arr[5];
int (\*pa)(int, int) = add;
int (\*pa[4])(int, int) = {add, add2, add3, add4};
//调用
printf("%d", pa[1](2, 3));
return 0;
}
char\* my\_strcpy(char\* dest, const char\* src);
1.写一个函数指针pf能够指向my\_strcpy
char\* (\*pf)(char\*, const char\*);
2.写一个函数指针数组pfarr 能够存放4个my\_strcpy函数的地址
char\* (\*pfarr[4])(char\*, const char\*);

使用案例：

计算器(简略)

int add(int x, int y);
int sub(int x, int y);
int mul(int x, int y);
int div(int x, int y);
int main()
{
int input = 0;
int x = 0;
int y = 0;
int (\*pf[4])(int, int) = {0, add, sub, mul, div};
do{
printf("请选择");
scanf("%d", \*input);
printf("%d%d", &x, &y);
int ret = pf[input](x, y);
printf(ret);
}while(input);
}

**7.指向函数指针数组的指针**

指向函数指针数组的指针是一个指针 指针指向一个数组 数组的元素都是函数指针

int main()
{
int arr[10] = {0};
int (\*p)[10] = &arr; //取出数组的地址
int (\*pp)(int, int);
int (\*pfarr[4])(int, int); //pfarr是一个数组 函数指针的数组
int (\*(\*ppfarr)[4])(int, int) = &pfarr;
//ppfarr是一个数组指针 指针指向的数组有四个元素
// 指向的数组的每个元素的类型是一个函数指针
return 0;
}

**8.回调函数**

通过函数指针调用的函数 如果把函数的地址作为参数传递给另一个函数 当这个指针被用来调用其他所指的函数时 我们就说这是回调函数，回调函数不是由该函数的实现方直接调用 而是在特定的时间或者条件发生时由另外的一方调用的 用于该事件或条件进行响应

**9.指针和数组面试题解析**