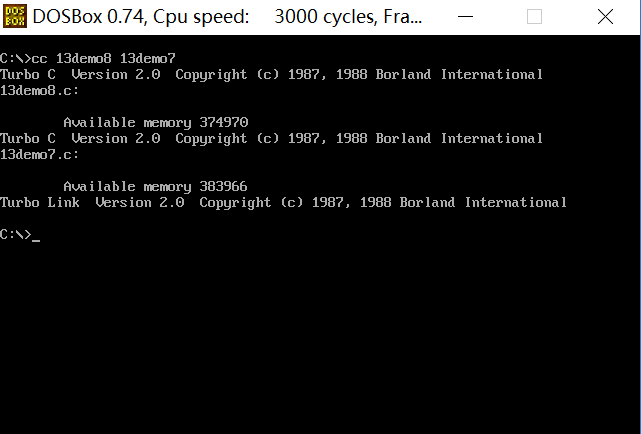
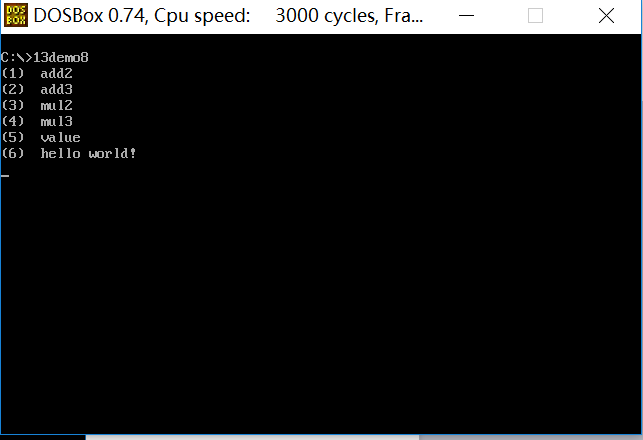
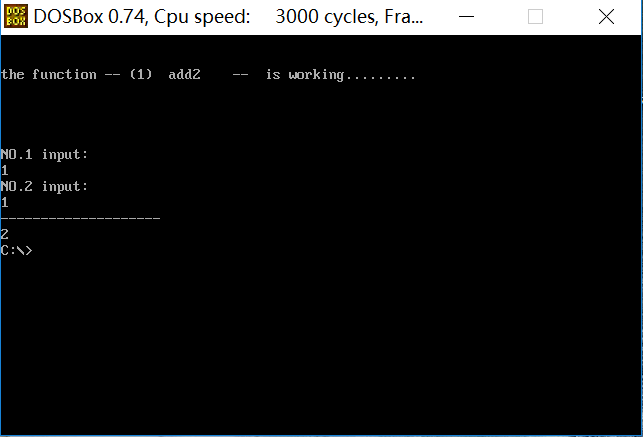
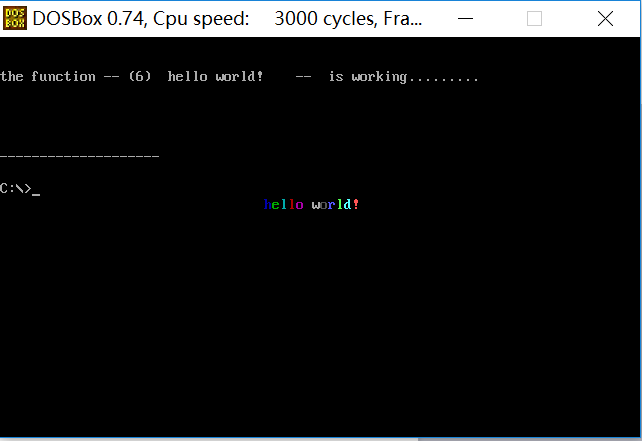
**\*\*1、重新搭建框架，要求按学长演示的来（框架），分析清楚客户需求！**









开发工具程序cc.c：

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

int main(int n,char \*\*arg)

{

int i;

char a[20] = "tcc -c ";

char b[80] = "tlink c0s main ";

char c[5] = ",";

char d[20] = " ,,cs.lib ";

char e[20] = "tcc -c ";

char f[5] = " ";

for(i = 0;arg[1][i]!='\0';i++)

{

if(arg[1][i] == '.')

{

arg[1][i] = '\0';

}

}

for(i = 0;arg[2][i]!='\0';i++)

{

if(arg[2][i] == '.')

{

arg[2][i] = '\0';

}

}

strcat(a,arg[1]);

system(a);

strcat(e,arg[2]);

system(e);

strcat(b,arg[1]);

strcat(b,f);

strcat(b,arg[2]);

strcat(b,c);

strcat(b,arg[1]);

strcat(b,d);

system(b);

return 0;

}

程序main.c:

#include "13demo6.h"

int a;

void menu(manage\* cpfmanage)

{

int b,i;

char ch;

for(a = 0;a < cpfmanage->fnum;a++)

{

b =strlen(cpfmanage->item[a].text);

if(b > 20 || b <0)

{

printf("error");

return;

}

printf("%s\n",cpfmanage->item[a].text);

}

do

{

ch = getch();

for(a = 0;cpfmanage->item[a].key != ch && a < cpfmanage->fnum ;a++);

}while(a == cpfmanage->fnum);

system("cls");

printf("\n\n");

printf("the function -- %s -- is working.........\n\n\n\n\n",cpfmanage->item[a].text);

for( i = 0;i<cpfmanage->item[a].pnum;i++ )

{

printf("NO.%d input: \n",i+1);

gets(str1[i]);

}

printf("--------------------\n");

}

void init()

{

menu(&fmanage);

}

main()

{

init();

change();

}

程序13demo6.h:

typedef struct

{

const char\* text;

char key;

void\* func;

int pnum;

}list;

typedef struct mg

{

list\* item;

void (\*doing)(struct mg\*);

int fnum;

}manage;

extern int a;

extern list flist[];

extern manage fmanage;

extern char str1[3][20];

void menu(manage\* cpfmanage);

extern void choose(manage\* cpfmanage);

void init();

extern void change();

extern int add2(int,int);

extern int add3(int,int,int);

extern int mul2(int,int);

extern int mul3(int,int,int);

extern int value(int);

extern void f1();

extern void putstr(char\*);

程序13demo7.c:

#include "13demo6.h"

int add2(int a,int b)

{

return a+b;

}

int add3(int a,int b,int c)

{

return a+b+c;

}

int mul2(int a,int b)

{

return a\*b;

}

int mul3(int a,int b,int c)

{

return a\*b\*c;

}

int value(int a)

{

return a;

}

void f1()

{

putstr("hello world!");

}

void putstr(char\* str)

{

int far\* scr =(int far\*)(0xb8000000+160\*10+33\*2);

int n;

for(n = 0;str[n];n++)

{

scr[n] = str[n]+(n+1)\*0x100;

}

}

程序13demo8.c:

#include "13demo6.h"

char str1[3][20];

char text1[20] = "(1) add2";

char text2[20] = "(2) add3";

char text3[20] = "(3) mul2";

char text4[20] = "(4) mul3";

char text5[20] = "(5) value";

char text6[20] = "(6) hello world!";

list flist[] = {

{ text1,'1', (void\*)add2,2 },

{ text2,'2', (void\*)add3,3 },

{ text3,'3', (void\*)mul2,2 },

{ text4,'4', (void\*)mul3,3 },

{ text5,'5', (void\*)value,1},

{ text6,'6', (void\*)f1 ,0 }

};

manage fmanage ={

flist,

menu,

sizeof(flist)/sizeof(list)

};

void change()

{

choose(&fmanage);

}

void choose(manage\* cpfmanage)

{

switch(cpfmanage->item[a].pnum)

{

case 1:

printf("%d",( \*(int (\*)(int))cpfmanage->item[a].func)(atoi(str1[0])) );

break;

case 2:

printf("%d",( \*(int (\*)(int,int) )cpfmanage->item[a].func)(atoi(str1[0]),atoi(str1[1])) );

break;

case 3:

printf("%d",( \*(int (\*)(int,int,int) )cpfmanage->item[a].func)(atoi(str1[0]),atoi(str1[1]),atoi(str1[2])) );

break;

case 0:

default:

( \*(void (\*)() )cpfmanage->item[a].func)() ;

break;

}

}

调用格式：

1. 将cc.c程序经tcc编译链接生成cc.exe开发工具程序；
2. 用“tcc -c”命令将main.c程序编译为main.obj文件；
3. 输入“cc 13demo8 13demo7”命令；
4. 执行“13demo8”。

文件的属性：

1. main.c文件编译为main.obj，不对外公开；
2. cc.c文件编译生成新开发工具cc.exe，不对外公开；
3. 13demo7.c文件由负责开发个性子函数，如二元加法、三元加法等函数的一名程序员进行统一地增删修改等操作，不对用户开放；
4. 13demo8.c文件由负责开发菜单新增函数选项并且逻辑处理调用几元参数的另一名程序员进行编写，不对用户开放；
5. 13demo6.h文件内容向用户开放。

**2、 void (\*\*f)(int,int)**

**void (\*f[4])(int,int)={add, sub, mul, div};**

**为什么不对，为什么main的二参不会出错？（二级指针和指针数组间的联系和区别？）**

数据存储空间不同；

Extern int (\*\*f)(int,int);

这个声明在对另一个文件中定义的同名的函数指针数组f变量在遇到类型冲突时进行了强制类型转换。

编写19demo1.c和19dmeo2.c看一下：

程序19demo1.c:

extern int i;

main()

{

p();

printf("%d\n",sizeof(i));

}

程序19demo2.c:

char i = 0;

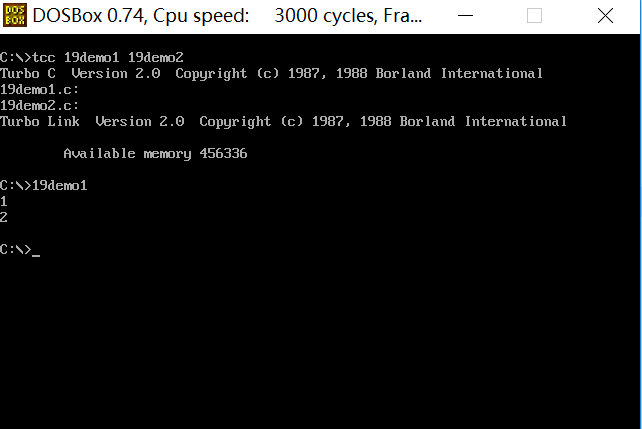
void p()

{

printf("%d\n",sizeof(i));

}

运行：



可以看到本来在19demo2.c中为char型的变量i在19demo1.c中被强制转换成了int型。

原本的mc.c中如果这样定义：

int (\*f[4])(int,int) = {add,sub,mul,div};

修改一下，看一下原本的值的变化：

程序19demo1.c：

extern int (\*\*f)(int,int);

main()

{

p();

printf("compare..............\n\n",f);

printf("%p\n",&f[0]);

printf("%p\n",&f[1]);

printf("%p\n",f);

}

程序19demo2.c:

int add(int a,int b)

{

return a+b;

}

int sub(int a,int b)

{

return a-b;

}

int (\*f[2])(int,int) = {add,sub};

void p()

{

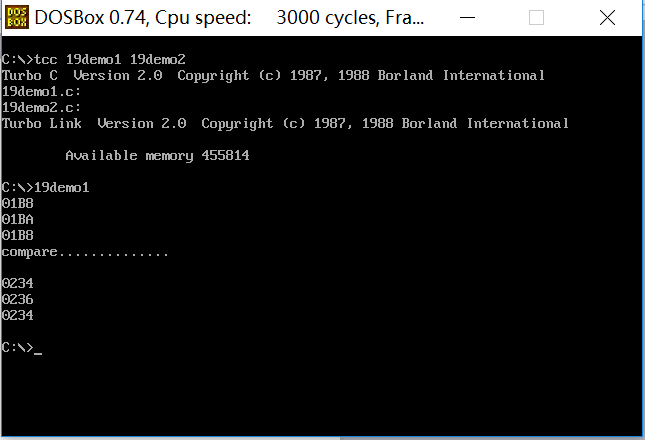
printf("%p\n",&f[0]);

printf("%p\n",&f[1]);

printf("%p\n",f);

}

运行结果：

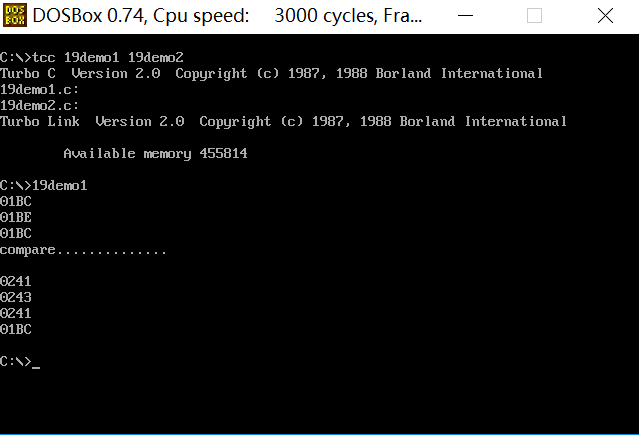


这里可以看到，数值已经发生了变化。

但是数值是怎样变化的呢？

在19demo1.c中加一句“ printf("%p\n",&f) ”

运行：



可以看到19demo1.c中的&f的数值和19demo2.c中的&f[0]和f的数值是一样的。

也就是强制类型转换的时候发生了地址的错位。如何避免呢？当然正常情况下是建立与之相匹配的extern外部声明，不会出现这种情况的。