**c语言综合研究13研讨会报告**

1. double型转int型时，运算到最后一位，为什么会不准？

之前学过，如果两个double型变量，比如“double a; double b;”,两个变量在比较值的大小的时候，一般不直接写成 “a == b”这类的形式，而是“ abs(a - b) < 0.11223”之类的。原本以为这样写是为了显得很精确的样子。这样看来，是不是double类型应用的局限性。

1. 针对拓展的时候只需要拓展唯一个性，基于研究13的数据组织方式，实现一个可显示并处理多种类型函数的程序。

如下：

"(1) add";

"(2) sub";

"(3) mul";

"(4) div";

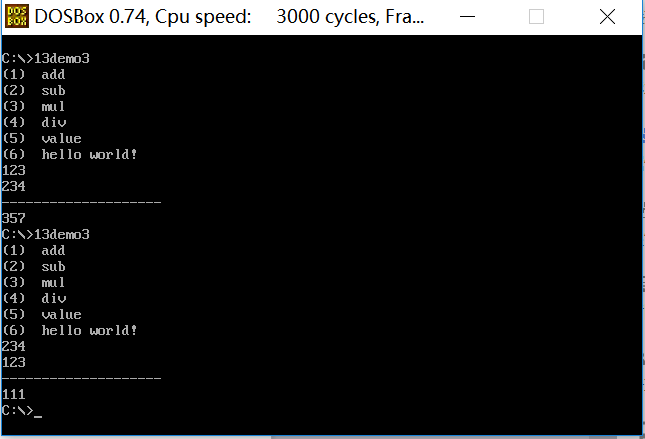
"(5) value";

"(6) hello world!";

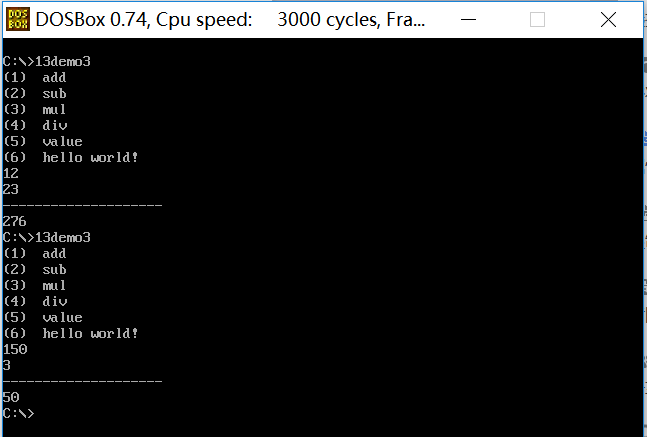
设计6个功能函数，前4个分别为二目运算的加减乘除，返回值,形参均为int类型；5号是返回值函数，单个形参，返回值，形参均为int类型；6号是显示字符串函数，返回值，形参均为void类型。

运行：

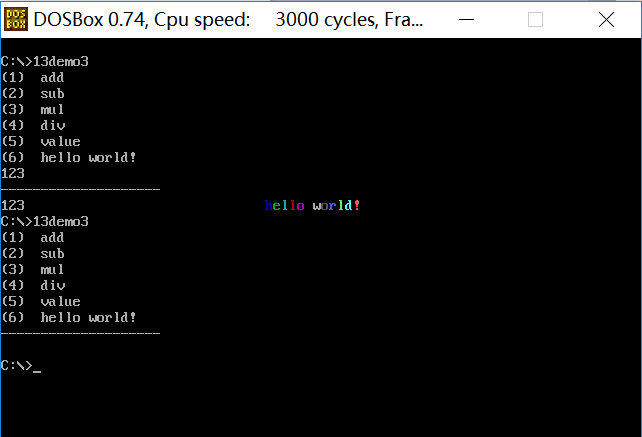
1加法、2减法：



3乘法、4除法：



5返回值、6显示：



13demo3.h:

typedef union{

void (\*setshow)();

int (\*setret)(int);

int (\*setadd)(int,int);

}funcType;

typedef struct c{

char\* text;

char key;

int can;

funcType choice;

}ItemType;

void menu(ItemType\* ,int);

char str1[3][20];

void menu(ItemType\* item ,int n)

{

int a,b,i;

char ch;

for(a = 0;a < n;a++)

{

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

if(b > 20 || b <0)

{

printf("error");

return;

}

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

}

do

{

ch = getch();

for(a = 0;item[a].key != ch && a < n ;a++);

}while(a == n);

for( i = 0;i<item[a].can;i++ )

{

gets(str1[i]);

}

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

switch(item[a].can)

{

case 1:

printf("%d",item[a].choice.setret(atoi(str1[0])));

break;

case 2:

printf("%d",item[a].choice.setadd(atoi(str1[0]),atoi(str1[1])) );

break;

case 0:

default:

item[a].choice.setshow() ;

break;

}

}

13demo3.c:

#include "13demo3.h"

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

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

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

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

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

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

int add(int,int);

int sub(int,int);

int mul(int,int);

int div(int,int);

int value(int);

void f1();

void putstr(char\*);

ItemType item[]={

{ text1,'1',2, {add} },

{ text2,'2',2, {sub} },

{ text3,'3',2, {mul} },

{ text4,'4',2, {div} },

{ text5,'5',1, {value }},

{ text6,'6',0, {f1 } }

};

main()

{

menu(item,6);

}

int add(int a,int b)

{

return a+b;

}

int sub(int a,int b)

{

return a-b;

}

int mul(int a,int b)

{

return a\*b;

}

int div(int a,int b)

{

return a/b;

}

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;

}

}

分析：封装得不好，增加新个性比如3目运算的时候还得修改switch-case的逻辑结构。还得找更好的处理多种函数返回值与参数类型及个数不同的封装问题的方法。

1. 对12、13的程序设计思想的理解，比如为什么要用结构体，用结构体的意义。

结构体，首先它可以将多种不同类型的数据进行打包，如果我们处理的数据都是同种类型，数组或许也可以选择。而谈到共性的话，一个程序中，对于多个对象分析之后，可能得出多个共性，而这些共性分别适合通过不同的数据类型来处理，所以从这一点看，结构体初步符合要求。

不过结构相对于数组而言还有一些自己的独特点：结构的大小固定，结构中的数据都有自己的名字，而且我如果像数组读取数组元素那样读取结构体元素，不报错都难，但是它有“.”运算符，搭配数据名字，这下数据名字可有用武之地了。数组变量还是地址，结构体变量不是。数组的数组是二维数组，但是结构也可以嵌套结构。

但是在定义的时候，我们也是只需要保证数据按照他们在结构中定义的顺序就行。

然后我们如果在函数中用到结构体指针传递数据，就好像把所有需要的东西都打包发往同一个目的地，这样省去了一个个传送，而且看起来还很有规则。而且回到12、13的程序中，把数据放在结构中传递的时候，我们修改结构的内容时，新增一个共性数据时，并不需要修改使用它的函数。

肯定还要为下一研究的面向对象进行铺垫。