mxw c语言测试框架

主函数文件

```
/*************************************************************************
***
        > File Name: main.cpp
        > Author:
        > Mail:
        > Created Time: Thu Dec 31 13:59:31 2020
 ************************************************************************
**/
#include <mxwgoog/test.h>

int add(int a, int b) {
    return a + b;
}

TEST(TESTfunc, add) {
    EXPECT_EQ(add(2, 0), 2);
    EXPECT_NE(add(3, 7), 10);
    EXPECT_EQ(add(2, 5), 6);
    EXPECT_GE(add(2, 8), 9);
}

TEST(TESTfunc, add2) {
    EXPECT_EQ(add(9, 0), 2);
    EXPECT_LT(add(3, 7), 10);
    EXPECT_GT(add(1, 5), 6);
    EXPECT_EQ(add(2, 8), 10);
}

TEST(TEST, funcadd) {
    EXPECT_EQ(add(2, 0), 2);
    EXPECT_LE(add(3, 7), 10);
    EXPECT_EQ(add(2, 5), 7);
    EXPECT_EQ(add(2, 7), 9);
}

int main(){
    return RUN_ALL_TESTS();
}
```

头文件

```
/*************************************************************************
***
> File Name: test.h
> Author:
> Mail:
> Created Time: Thu Dec 31 14:02:04 2020
*************************************************************************
*/
#ifndef _TEST_H
#define _TEST_H

#include <mxwgoog/linklist.h>
#include <stdlib.h>
#include <math.h>
#include <stdio.h>
#include <string.h>

#define COLOR(a, b) "\033[" #b "m" a "\033[0m"
#define COLOR_HL(a, b) "\033[1;" #b "m" a "\033[0m"

#define GREEN(a) COLOR(a, 32)
#define RED(a) COLOR(a, 31)
#define BLUE(a) COLOR(a, 34)
#define YELLOW(a) COLOR(a, 33)
#define PURPLE(a) COLOR(a, 35)

#define GREEN_HL(a) COLOR_HL(a, 32)
#define RED_HL(a) COLOR_HL(a, 31)
#define BLUE_HL(a) COLOR_HL(a, 34)
#define YELLOW_HL(a) COLOR_HL(a, 33)
#define PURPLE_HL(a) COLOR_HL(a, 35)

#define TEST(a, b)\
void a##_haizei_##b();\
__attribute__((constructor))\
void add##_haizei_##a##_haizei_##b() {\
    add_function(a##_haizei_##b, #a "_haizei_" #b);\
}\
void a##_haizei_##b()

#define Type(a) _Generic((a), \
    int : "%d",\
```

```
    char : "%c",\
    float : "%f",\
    double : "%lf",\
    char * : "%s",\
    long long : "%lld",\
    const char * : "%s"\
)

#define Print(a, color) {\
    char *str;\
    str = (char *)malloc(sizeof(char) * 1000);\
    sprintf(str, color("%s"), Type(a));\
    printf(str, a);\
    free(str);\
}

#define EXPECT(a, b, comp){\
   __typeof(a) _a = (a);\
   __typeof(b) _b = (b);\
   haizei_test_info.total += 1;\
   if (_a comp _b) haizei_test_info.success += 1;\
   else {\
        printf(YELLOW_HL("\t%s : %d : failure\n"), __FILE__,
__LINE__);\
        printf(YELLOW_HL("\t\texpect : " #a " " #comp " " #b "\n\t\t"
"actula : "));\
        Print(a,YELLOW_HL)\
        printf(YELLOW_HL("  vs  "));\
        Print(b, YELLOW_HL);\
        printf("\n");\
   }\
   printf(GREEN("[-----------]"));\
   printf(BLUE_HL("%s %s %s") " %s \n", #a, #comp, #b, _a comp _b ?
GREEN("True") : RED("False"));\
}

#define EXPECT_EQ(a, b) EXPECT(a, b, ==)
#define EXPECT_NE(a, b) EXPECT(a, b, !=)
#define EXPECT_LT(a, b) EXPECT(a, b, <)
#define EXPECT_LE(a, b) EXPECT(a, b, <=)
#define EXPECT_GT(a, b) EXPECT(a, b, >)
#define EXPECT_GE(a, b) EXPECT(a, b, >=)

typedef void (*TestFuncT)();

```

```
typedef struct Function {
    TestFuncT func;
    const char *str;
    struct LinkNode p;
} Function;

struct FunctionInfo {
    int total, success;
};

extern struct FunctionInfo haizei_test_info;
int RUN_ALL_TESTS();
void add_function(TestFuncT, const char *);

#endif
```

```
/*************************************************************************
***
        > File Name: mxwgoog/linklist.h
        > Author:
        > Mail:
        > Created Time: Tue Jan  5 15:54:51 2021
 *************************************************************************
**/

#ifndef _LINKLIST_H
#define _LINKLIST_H

#define offset(T, name) (long long)(&(((T *)(NULL))->name))
#define Head(p, T, name) (T *)((char *)(p) - offset(T, name))

struct LinkNode {
    struct LinkNode *next;
};

#endif
```

源文件

```
#include <mxwgoog/linklist.h>
#include <mxwgoog/test.h>

int func_cnt = 0;
Function func_head, *func_tail = &func_head;
```

```
struct FunctionInfo haizei_test_info;
int RUN_ALL_TESTS() {
    for (struct LinkNode *p = func_head.p.next; p; p = p->next) {
        Function *func = Head(p, Function, p);
        printf(GREEN("[====RUN====]")  RED_HL("%s\n"), func->str);
        haizei_test_info.total = 0, haizei_test_info.success = 0;
        func->func();
        double rate = 100.0 * haizei_test_info.success / haizei_test_info.total;
        printf(PURPLE("[  "));
        if (fabs(rate - 100 ) < 1e-7) {
            printf(GREEN_HL("%6.2lf%%"), rate);
        } else {
            printf(RED_HL("%6.2lf%%"), rate);
        }
        printf(PURPLE("  ] ") GREEN_HL(" total : %d    success : %d\n"),
               haizei_test_info.total,
               haizei_test_info.success
         );
    }
    return 0;
}

void add_function(TestFuncT func, const char *str) {
    Function *temp = (Function *)calloc(1, sizeof(Function));
    temp->func = func;
    temp->str = strdup(str);
    func_tail->p.next = &(temp->p);
    func_tail = temp;
    return ;
}
```

执行命令块

```
.PHONY:clean
all: mxwgoog/main.o mxwgoog/test.o
        gcc -I./ mxwgoog/main.o mxwgoog/test.o -o ./mxwgoog/xianshi.out
mxwgoog/main.o: mxwgoog/main.c mxwgoog/test.h
        gcc -I./ -c mxwgoog/main.c -o mxwgoog/main.o
mxwgoog/test.o: mxwgoog/test.c mxwgoog/test.h mxwgoog/linklist.h
        gcc -I./ -c mxwgoog/test.c -o mxwgoog/test.o
clean:
        rm -rf ./mxwgoog/xianshi.out mxwgoog/main.o mxwgoog/test.o
```