

实验 2 数据表示和运算实验

一、实验目的

- 1 了解并学习计算机的数据表示方式，了解并学习计算机的算术运算方式，理解不同数据类型的运算属性。
- 2 了解并学习 gdb 的使用方法，并运用其进行内存、寄存器检查。

二、实验内容

1、在 32 位计算机中运行一个 C 语言程序，在该程序中出现了以下变量的初值，请在表格中填写它们对应的机器数（用十六进制表示）。在 gdb 里面可使用 x/1xw 查看 int/unsigned/float 的机器数，使用 x/1xh 查看 short/unsigned short 的机器数，使用 x/1xb 查看 char 的机器数，使用 x/1xg 查看 double 的机器数：

- (1) int x=-32768 (2) short y=522 (3) unsigned z=65530 (4) char c='@'
(5) float a=-1.1 1 (6) double b=10.5 (7) float u = 123456.789e4 (8) double
v= 123456.789e4

变量	x	y	z	c
机器数				
变量	a	b	u	v
机器数				

运行下面的代码验证输出是否与 gdb 查看的结果一致：

```
#include <stdio.h>

int main() {
    int x = -32768;
    short y = 522;
    unsigned z = 65530;
    char c = '@';
    float a = -1.1;
    double b = 10.5;
    float u = 123456.789e4;
    double v = 123456.789e4;

    printf("+++++++Machine value+++++++\n");
    printf("x = 0x%x\n", x);
    printf("y = 0x%hx\n", y);
    printf("z = 0x%x\n", z);
    printf("c = 0x%hhx\n", c);
    printf("a = 0x%x\n", *(unsigned *)&a);
    printf("b = 0x%llx\n", *(unsigned long long *)&b);
    printf("u = 0x%x\n", *(unsigned *)&u);
    printf("v = 0x%llx\n", *(unsigned long long *)&v);
    printf("+++++++Real value+++++++\n");
```

```

printf("x = %d\n", x);
printf("y = %hd\n", y);
printf("z = %u\n", z);
printf("c = %c\n", c);
printf("a = %f\n", a);
printf("b = %f\n", b);
printf("u = %f\n", u);
printf("v = %f\n", v);
}

```

2、使用命令 `gcc -ggdb swap.c -o swap` 编译下面的 `swap.c` 代码，完成后面的实验

```

void xor_swap(int *x, int *y){
    *y=*x ^ *y; /* 第一步 */
    *x=*x ^ *y; /* 第二步 */
    *y=*x ^ *y; /* 第三步 */
}

int main() {
    int a = 1;
    int b = 2;
    xor_swap(&a, &b);
}

```

1) 使用 `gdb` 命令查看程序变量的取值，填写下面两个表格：

a 的存放地址(&a)	b 的存放地址(&b)	x 的存放地址(&x)	y 的存放地(&y)

执行步数	x 的值(机器值, 用十六进制)	y 的值(机器值, 用十六进制)	*x 的值(程序中的真值, 用十进制)	*y 的值(程序中的真值, 用十进制)
第一步前				
第一步后				
第二步后				
第三步后				

2) 运行下面的 `reverse.c`，并说明输出这种结果的原因，修改代码以得到正确的逆序数组

```

#include <stdio.h>
void xor_swap(int *x, int *y) {
    *y=*x ^ *y; /* 第一步 */
    *x=*x ^ *y; /* 第二步 */
}

```

```

    *y=*x ^ *y; /* 第三步 */
}
void reverse_array(int a[], int len) {
    int left, right=len-1;
    for (left=0; left<=right; left++, right--)
        xor_swap(&a[left], &a[right]);
}
int main() {
    int a[] = {1,2,3,4,5,6,7};
    reverse_array(a, 7);
    int i;
    for(i = 0; i < 7; ++i)
        printf("%d ", a[i]);
    printf("\n");
}

```

3、编译并运行下面的程序，使用 **gdb** 指令查看变量的取值，解释语句输出为 **False** 的原因并填写在表格中。

```

#include <stdio.h>
#include <limits.h>
int main(){
    int x = INT_MAX;
    float xf = x;
    double xd = x;
    printf("+++++++True or False+++++++\n");
    printf("x==(int)xd %s\n",x==(int)xd?"True":"False");//语句一
    printf("x==(int)xf %s\n",x==(int)xf?"True":"False");//语句二

    float p1 = 3.141592653;
    float p2 = 3.141592654;
    printf("p1!=p2 %s\n",p1!=p2?"True":"False");//语句三

    float f = 1.0e20;
    double d = 1.0;
    double result1 = d+(f-f);
    double result2 = (d+f)-f;
    printf("result1==d %s\n",result1==d?"True":"False");//语句四
    printf("result2==d %s\n",result2==d?"True":"False");//语句五
}

```

	输 出 True/False	原因
语句一		

语句二		
语句三		
语句四		
语句五		

4、观察下面 data_rep.c 程序的运行：

```
int main() {
    char x  = 0x66;
    char y = 0x39;
    char x_bit_not = ~x;
    char x_not = !x;
    char x_bit_and_y = x & y;
    char x_and_y = x  && y;
    char  x_bit_or_y = x | y;
    char x_or_y = x || y;

    int x1 = (1<<31)-1;
    int y1 = 1;
    int sum_x1_y1 = x1 + y1;
        int diff_x1_y1 = x1 - y1;
        int diff_y1_x1 = y1 - x1;

    unsigned int x2 = (1<<31)-1;
    unsigned int y2 = 1;
        unsigned int sum_x2_y2 = x2 + y2;
        unsigned int diff_x2_y2  = x2 - y2;
        unsigned int diff_y2_x2  = y2 - x2;
}
```

1) 使用命令 `gdbtui data_rep` 进入 gdb 的 TUI 调试模式，之后分别输入命令：`layout asm` 和 `layout regs`，再输入命令 `start` 启动程序，然后使用 `si` 命令进行单步运行。请在单步运行过程中完成下面的表格（如果 TUI 模式有问题就直接用 `gdb`，忽略掉前面的提示即可）：

	机器数 (十六进制)	真值 (十进制)		机器数 (十六进制)	真值 (十进制)
x			y		
~x			!x		
x & y			x && y		
x y			x y		

	机器数 (十六进制)	真值 (十进制)	OF	SF	CF	AF
x1						
y1						
sum_x1_y1						
diff_x1_y1						
diff_y1_x1						
x2						
y2						
sum_x2_y2						
diff_x2_y2						
diff_y2_x2						

2) 写出上面表格中每个标识位变化的原因，可直接在上表中注明。

提交要求：

请在规定时间内提交一个以学号为名的压缩文件，如 151220000.zip 到课程网站（注意修改学号和压缩格式，不接受过期提交）。压缩包内部应该是一个目录。

压缩文件解压后获得目录内容如下（注意文件名大小写和每一个文件的提交要求）：

151220000

|----reverse.c

|----report.pdf // report 中应包含实验中的所有表格