

重庆大学

学生实验报告

实验课程名称 操作系统

开课实验室 DS1503

学 院 大数据与软件学院 年级 软件工程 专业班
01

学 生 姓 名 学 号

开 课 时 间 2024 至 2025 学年第二 学期

总 成 绩	
教师签名	

《操作系统》实验报告

开课实验室：

2025 年 3 月 1 日

学院	大数据与软件学院	年级、专业、班	2023/软件工程 /01	姓名	成绩	
课程名称	操作系统	实验项目名称	实验二：线程的创建		指导教师	刘寄
教师评语	<p style="text-align: right;">教师签名： 年 月 日</p>					

一、实验目的

实现线程创建和简单图形化实例

二、实验内容

1.

- 随机生成N组非负整数列表，然后创建N个线程，分别用N种不同的排序算法对列表进行排序
 - N必须大于2
 - 如何生成随机数?
 - Step1-播种: `void srand(uint32_t seed)`
 - seed是随机数的种子，建议用实验（一）中实现的系统调用“`time_t time(time_t *loc)`”
 - » `srand(time(NULL))`
 - Step2-生成: 多次调用“`int rand()`”获得随机数

2.

- 进入图形模式，沿垂直方向把屏幕分成N个区域，每个排序线程用一个区域，动态显示排序过程。
 - 如何进入图形模式?
 - 调用`init_graphic(int mode)`
 - » mode=0x143
 - 如何获取屏幕的分辨率?
 - 水平: `g_graphic_dev.XResolution`, 垂直: `g_graphic_dev.YResolution`
 - 如何打点?
 - `void setPixel(int x, int y, COLORREF cr);`
 - » (x, y)是点坐标
 - » cr是颜色，用宏定义RGB(r,g,b)生成，其中r,g,b的取值范围都是0-255
 - 如何从cr中取出r,g,b? 用`getXValue(cr)`, 其中X=R,G,B
 - 如何画线?
 - `void line(int x1, int y1, int x2, int y2, COLORREF cr);`
 - 如何退出图形模式?
 - `int exit_graphic();`

三、使用仪器、材料

虚拟机，编译器

三、实验过程原始记录(数据、图表、计算等)

测试线程调用（用数组记录排序过程）

使用选择，插入，冒泡排序，将线程 id 加入随机种子避免生成相同数组

```
59 task #0: Loading a.out...Done
60 task #0: Creating first user task...task #1: I'm the first user task(pv=0x
61 78) !
62 This is task Bubbling sorting with tid=2
63 6 7 25 25 33 52 53 64 67 79
64
65 This is task Select Sort with tid=3
66 2 24 25 32 38 47 68 69 79 82
67
68 This is task Insert Sort with tid=4
69 5 13 30 31 36 39 69 77 78 96
70
71 gcc -m32 -nostdlib -nostartfiles -nodefaultlibs -Wl,-Map,a.map -sta
72 per.o vm86call.o graphics.o main.o lib/sysconf.o lib/math.o lib/std
73 ib/string.o ../lib/memcpy.o ../lib/memset.o ../lib/snprintf.o lib/m
74 make[1]: Leaving directory `/d/2025_02_27_expenv/expenv/epos/userap
75 if [ ! -s hd.img ]; then base64 -d hd.img.bz2.txt | bunzip2 >hd.img
```

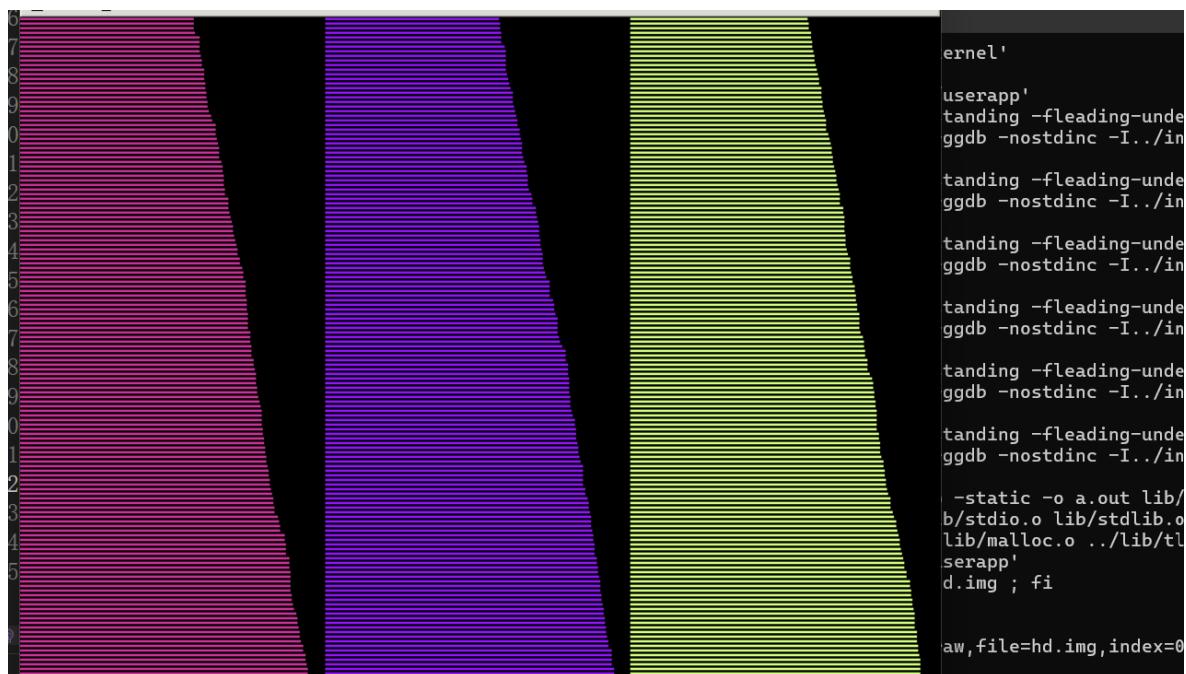
测试图形化渲染（边界和坐标系正方向）

原点位于左上角，右为 x 正方向，下为 y 正方向

调整刷新率（避免闪黑屏）

先渲染一帧然后刷新每条线

五、实验结果及分析



源代码：

```
void tsk_b(void* pv)
{
    printf("This is task Bubbling sorting with tid=%d\r\n", task_getid());
    //生成随机数组:
    unsigned int seed = (unsigned int)task_getid() + (unsigned int)time(NULL);
    srand(seed); // 将线程 id 作为种子的一部分避免生成数组一样
    int i;
```

```

int j;
int k;
for (i = 0; i < 150; i++) {
    a[i] = (rand() % 106) + 150;//为了可视化明显，生成 150~255 之间的数
}
// 冒泡排序
for (i=0; i < 149; i++)
{
    for (j = 0; j < 149 - i; j++)
    {
        if (a[j] > a[j + 1])
        {
            int t = a[j];
            a[j] = a[j + 1];
            a[j + 1] = t;
        }
    }
    for (k = 0; k < 150; k++)
    {
        b[k + i * 150] = a[k];
    }
}
// 输出排序结果
for (i = 0; i < 150; i++)
{
    printf("%d ", a[i]);
}

task_exit(0);//不能直接 return，必须调用 task_exit
}

void tsk_s(void* pv)
{
    printf("\n");
    printf("\n");
    printf("This is task Select Sort with tid=%d\r\n", task_getid());
    //生成随机数组：
    unsigned int seed = (unsigned int)task_getid() + (unsigned int)time(NULL);
    srand(seed);
    int i;
    int j;
    int k;
    for (i = 0; i < 150; i++) {
        a[i] = (rand() % 106) + 150;
    }
    // 选择排序
}

```

```

for (i = 0; i < 150; i++) {
    int index = i;

    // 在未排序部分中查找最小值
    for (j = i + 1; j < 150; j++)
    {
        if (a[j] < a[index]) index = j;
    }

    // 将最小值交换到已排序部分的末尾
    if (index != i)
    {
        int temp = a[i];
        a[i] = a[index];
        a[index] = temp;
    }

    for (k = 0; k < 150; k++)
    {
        s[k + i * 150] = a[k];
    }
}

// 输出排序结果

for (i = 0; i < 150; i++)
{
    printf("%d ", a[i]);
}

task_exit(0); //不能直接 return, 必须调用 task_exit
}

void tsk_i(void* pv)
{
    printf("\n");
    printf("\n");
    printf("This is task Insert Sort with tid=%d\r\n", task_getid());
    //生成随机数组:
    unsigned int seed = (unsigned int)task_getid() + (unsigned int)time(NULL);
    srand(seed);
    int i;
    int k;
    for (i = 0; i < 150; i++) {
        a[i] = (rand() % 106) + 150;
    }

    // 插入排序
    for (i = 1; i < 150; i++) {
        int cur = a[i];

```

```

int j = i - 1;

while (j >= 0 && a[j] > cur) {
    a[j + 1] = a[j];
    j--;
}
a[j + 1] = cur;
for (k = 0; k < 150; k++)
{
    in[k + (i-1) * 150] = a[k];
}
}

// 输出排序结果

for (i = 0; i < 150; i++)
{
    printf("%d ", a[i]);
}

task_exit(0); //不能直接 return, 必须调用 task_exit
}

unsigned char* stack_b;
unsigned int stack_size = 1024 * 1024;
stack_b = (unsigned char*)malloc(stack_size);
int tid_b;
tid_b = task_create(stack_b + stack_size, &tsk_b, (void*)0);

unsigned char* stack_s;
stack_s = (unsigned char*)malloc(stack_size);
int tid_s;
tid_s = task_create(stack_s + stack_size, &tsk_s, (void*)0);

unsigned char* stack_i;
stack_i = (unsigned char*)malloc(stack_size);
int tid_i;
tid_i = task_create(stack_i + stack_size, &tsk_i, (void*)0);

//图形模式显示排序过程
init_graphic(0x0143); //800x600
//休眠时间
struct timespec req = {
    .tv_sec = 0,          // 秒
    .tv_nsec = 500000000  // 纳秒 (500,000,000 ns = 0.5 秒)
};
int i;

```

```
int j;
//测试显示化
//先显示一帧，画两条线代表一个数据，后续刷新线条避免屏幕闪黑
for (i = 0; i < 150; i++)
{
    line(0, 4 * i, b[i], 4 * i, RGB(153, 50, 204));
    line(0, 4 * i + 1, b[i], 4 * i + 1, RGB(153, 50, 204));
}///冒泡的大数组上限是 148

for (i = 0; i < 150; i++)
{
    line(265, 4 * i, s[i] + 265, 4 * i, RGB(255, 20, 147));
    line(265, 4 * i + 1, s[i] + 265, 4 * i + 1, RGB(255, 20, 147));
}

} //选择的大数组上限是 149

for (i = 0; i < 150; i++)
{
    line(265 * 2, 4 * i, in[i] + 265 * 2, 4 * i, RGB(127, 255, 212));
    line(265 * 2, 4 * i + 1, in[i] + 265 * 2, 4 * i + 1, RGB(127, 255, 212));
}

} //插入的大数组上限是 148
for (j = 1; j < 149; j++)
{
    for (i = 0; i < 150; i++)
    {
        if (j < 148)
        {
            line(0, 4 * i, 265, 4 * i, RGB(0, 0, 0));
            line(0, 4 * i + 1, 265, 4 * i + 1, RGB(0, 0, 0));
        }
        line(0, 4 * i, b[i + j * 150], 4 * i, RGB(153, 50, 204));
        line(0, 4 * i + 1, b[i + j * 150], 4 * i + 1, RGB(153, 50, 204));
    }
} //冒泡的大数组上限是 148

for (i = 0; i < 150; i++)
{
    if (j < 148)
    {
        line(265, 4 * i, 2*265, 4 * i, RGB(0, 0, 0));
        line(265, 4 * i + 1, 2*265, 4 * i + 1, RGB(0, 0, 0));
    }
    line(265, 4 * i, s[i + j * 150] + 265, 4 * i, RGB(255, 20, 147));
}
```

```

        line(265, 4 * i + 1, s[i + j * 150] + 265, 4 * i + 1, RGB(255, 20, 147));

    } //选择的大数组上限是 149

    for (i = 0; i < 150; i++)
    {

        if (j < 148)
        {
            line(265 * 2, 4 * i, 265 * 3, 4 * i, RGB(0, 0, 0));
            line(265 * 2, 4 * i + 1, 265 * 3, 4 * i + 1, RGB(0, 0, 0));
        }
        line(265 * 2, 4 * i, in[i + j * 150] + 265 * 2, 4 * i, RGB(127, 255, 212));
        line(265 * 2, 4 * i + 1, in[i + j * 150] + 265 * 2, 4 * i + 1, RGB(127, 255, 212));
    } //插入的大数组上限是 148
    nanosleep(&req, NULL);
}

//最后把选择排序的最后一帧补上
for (i = 0; i < 150; i++)
{
    if (j < 148)
    {
        line(265, 4 * i, 2 * 265, 4 * i, RGB(0, 0, 0));
        line(265, 4 * i + 1, 2 * 265, 4 * i + 1, RGB(0, 0, 0));
    }
    line(265, 4 * i, s[i + 149 * 150] + 265, 4 * i, RGB(255, 20, 147));
    line(265, 4 * i + 1, s[i + 149 * 150] + 265, 4 * i + 1, RGB(255, 20, 147));

} //选择的大数组上限是 149

```

实验报告打印格式说明

1. 标题：三号加粗黑体
2. 开课实验室：5 号加粗宋体
3. 表中内容：
 - (1) 标题：5 号黑体
 - (2) 正文：5 号宋体

4. 纸张: 16 开 ($20\text{cm} \times 26.5\text{cm}$)

5. 版芯

上距: 2cm

下距: 2cm

左距: 2.8cm

右距: 2.8cm

说明: 1、“年级专业班”可填写为“00 电子 1 班”，表示 2000 级电子工程专业第 1 班。

2、实验成绩可按五级记分制（即优、良、中、及格、不及格），或者百分制记载，若需要将实验成绩加入对应课程总成绩的，则五级记分应转换为百分制。