《数据结构与算法》实验报告

实验六：并发程序设计

教　师：潘晔

学　生：梁书恺

学　号：2022040906023

时　间：11.20

地　点：科 B119

一、ex1-1

1. 问题分析：

1）题意理解

利用 Windows 平台的 api，进行进程的创建、销毁，线程的创建，并控制线程同步。

2）数据结构设计

无

3）关键算法思路（画图或伪语言说明）

```
if (!CreateProcess(NULL, szCommandLine, 0, 0, 0, 0, 0, 0, &si, &pi))
{
    fprintf(stderr, "Createprocess Failed \n");
    return 0;
}
```

```
if (ResumeThread(pi.hThread) != -1) // 调用激活线程的函数
    cout << "激活进程成功" << endl;
else
    cout << "激活失败" << endl;
```

```
if (TerminateProcess(pi.hProcess, 0) != 0) // 调用终止线程的函数
    cout << "销毁进程成功" << endl;
else
    cout << "销毁失败" << endl;
```

```
// 创建一个信号量，初值为 0，最大为 1
HANDLE hSem = CreateSemaphore(NULL, 0, 1, NULL);
```

```
DWORD WINAPI Thread2(LPVOID lpParam)
{
    int count = 0;
    HANDLE hSem = (HANDLE)lpParam;
    while (!isStop)
    {
        if (count % 20 == 0)
        {
            WaitForSingleObject(hSem, INFINITE); // 申请信号量
            count = 0;
        }
        printf("T2:%2d ", count + 1);
        count++;
        Sleep(100);
    }
    return 0;
}
```

```
DWORD WINAPI Thread3(LPVOID lpParam)
{
    int count = 0;
    HANDLE hSem = (HANDLE)lpParam;
    while (!isStop)
    {
        if (count % 30 == 0)
        {
            ReleaseSemaphore(hSem, 1, NULL); // 释放信号量
            count = 0;
        }
        printf("T3:%2d ", count + 1);
        count++;
        Sleep(100);
    }
    return 0;
}
```

2. 测试数据与运行结果截图：

请输入要选择的操作：
0:退出
1:创建进程
2:挂起进程
3:激活进程
4:销毁进程
5:启动两个一样的线程
6:启动两个不同线程
7:停止线程

输入：1
记事本软件被打开

输入：2
输出：挂起进程成功

输入：3
输出：激活进程成功

输入：4
输出：销毁进程成功

输入：5
输出：22 11 22 11 11 22
11 22 11 22 22 11 11
22 11 22 11 22 11 22
11 22 11 22 11 22 11
22 11 22 11 22 11 22
11 22 11 22 11 22 11
22 11 22 11 22 11 22
11 22 11 22 11 22 11
22 11 22 11 22 11 22
11 22 11 22 11 22 11
11 22 11 22 11 22 11
22 11 22 11 22 11 22 …

输入：6
输出：T3: 1 T2: 1 T2: 2 T3: 2 T2: 3 T3: 3 T3: 4 T2: 4
T3: 5 T2: 5 T3: 6 T2: 6 T2: 7 T3: 7 T2: 8 T3: 8 T2: 9
T3: 9 T3:10 T2:10 T3:11 T2:11 T3:12 T2:12 T2:13 T3:13
T3:14 T2:14 T3:15 T2:15 T2:16 T3:16 T3:17 T2:17 T3:18
T2:18 T2:19 T3:19 T2:20 T3:20 T3:21 T3:22 T3:23 T3:24
T3:25 T3:26 T3:27 T3:28 T3:29 T3:30 T3: 1 T2: 1 T3: 2
T2: 2 T3: 3 T2: 3 T2: 4 T3: 4 T2: 5 T3: 5 T3: 6 T2: 6
T2: 7 T3: 7 T2: 8 T3: 8 T3: 9 T2: 9 T2:10 T3:10 T3:11 …

输入：7
线程停止，回到选择输入界面

3. **上机时遇到的问题（可分为编译问题和逻辑问题）**
   ① **问题现象：** 终止线程失败 **原因：**传入参数错误； **解决办法：**将原本的 pi.hThread 参数改为 pi.hProcess。
   ② **问题现象：** 输入 7 后线程不停止输出 **原因：**主函数逻辑有误，同时线程中没有加入停止判断； **解决办法：**修改主函数逻辑，并在线程中加入停止判断。

4. **程序代码**
   process.cpp

小结体会

本次上机实验和以往的有较大的不同，这次的程序给定了框架，需要补充其中的内容，这需要仔细阅读文档中的说明。在创建进程的时候，函数的参数较多，需要仔细对照每个参数。终止进程时，需要注意传入的参数是进程句柄，而不是线程句柄。