1. 进程是什么

- 是为正在运行的程序建立的一个管理实体
- 是一个具有一定独立功能的程序关于某个数据集合的一次运行活动
- 是操作系统进行资源分配和调度的一个独立单位

2. 进程表是什么?

- 进程表是存储进程状态信息的数据结构，包括标识信息、状态信息、控制信息。进程表是进程存在的唯一标识，是操作系统用来记录和刻画进程状态及环境信息的数据结构，是进程动态特征的汇集，也是操作系统掌握进程的唯一资料结构和管理进程的主要依据

3. 进程栈是什么?

- 进程运行时，系统分配给每个进程的栈空间。

4. 当寄存器的值已经被保存到进程表内，esp 应该指向何处来避免破坏进程表的值?

- 我们在进程调度模块中会用到堆栈，而寄存器被压到进程表之后，`esp` 是指向进程表某处的。接下来进行任何的堆栈操作，都会破坏掉进程表的值，从而在下一次进程恢复时产生严重的错误。为解决这个问题，必须将 `esp` 指向专门的内核栈区域。

5. tty是什么?

- tty,teletypewriter原为电传打字机的缩写,随着时间的发展变为了一个跟终端有关的混乱术语
- 在现代概念中,tty变成了Linux的一个子系统,通过 TTY 驱动程序在内核级别实现进程管理、行编辑和会话管理

6. 不同的tty为什么输出不同的画面在同一个显示器上?

- 不同 TTY 各有一个 CONSOLE，各个 CONSOLE 公用同一块显存的不同位置

7. 解释tty任务执行的过程?

- 在 TTY 任务中执行一个循环，这个循环将轮询每一个 TTY，处理它的事件，包括从键盘缓冲区读取数据、显示字符等内容。轮询到每一个 TTY 时：
  1.处理输入：查看其是否为当前 TTY。只有当某个 TTY 对应的控制台是当前控制台时，它才可以读取键盘缓冲区；
  2.处理输出：如果有要显示的内容则显示它

8. tty结构体中大概包括哪些内容?

```
#define TTY_IN_BYTES    256    /* tty input S size */

struct s_console;

/* TTY */
typedef struct {
    u32 in_buf[TTY_IN_BYTES];    /* TTY 输入缓冲区 */
    u32 *p_inbuf_head;        /* 指向缓冲区中下一个空闲位置 */
    u32 *p_inbuf_tail;        /* 指向键盘任务应处理的键值 */
    int inbuf_count;        /* 缓冲区中已经填充了多少 */

    struct s_console *p_console;
} TTY;
```

9. console结构体中大概包括哪些内容?

```c
/* CONSOLE */
typedef struct s_console {
    unsigned int current_start_addr;    /* 当前显示到了什么位置 */
    unsigned int original_addr;         /* 当前控制台对应显存位置 */
    unsigned int v_mem_limit;           /* 当前控制台占的显存大小 */
    unsigned int cursor;                /* 当前光标位置 */
    u8 color;
} CONSOLE;
```

10. 什么是时间片?

    时间片是分时操作系统分配给每个正在运行的进程微观上的一段 CPU 时间。

11. 结合实验代码解释什么是内核函数？ 什么是系统调用?
    - 内核函数是仅在内核态执行的函数，执行特权操作，不能被用户态程序直接调用
      系统调用是提供给用户态程序请求内核服务，实现特权操作的访问的接口
    - 看proto.h