操作系统与网络 2025 春期末考试回忆版

命题：乐航睿　　回忆：氢氰酸

1. 判断题 50 道，每个部分的都有不少（不要觉得文件讲的少就考的少），基础概念（50）
2. 单选题 10 道（10）
3. 有三个进程，A 在 0s 到达需要 100s 完成，B 在 10s 到达需要 45s 完成，C 在 50s 到达需要 15s 完成，如何调度使平均周转时间最短，算出最短的平均周转时间（5）
4. 简述进程与线程的概念和最主要区别（5）
5. 简述 TCP 和 UCP 的区别（5）
6. 简述上下文切换的过程（5）
7. 写注释（5）：

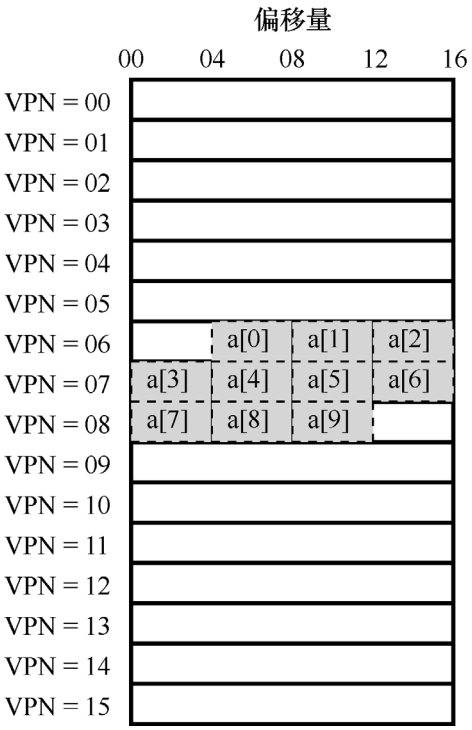
```
pthread_cond_wait(pthread_cond_t *c, pthread_mutex_t *m); //有什么用
pthread_cond_signal(pthread_cond_t *c); //有什么用
1 int done = 0;
2 pthread_mutex_t m = PTHREAD_MUTEX_INITIALIZER;    //有什么用
3 pthread_cond_t c = PTHREAD_COND_INITIALIZER;      //有什么用
4
5 void thr_exit() {
6 Pthread_mutex_lock(&m);   //为什么要这么做
7 done = 1;
8 Pthread_cond_signal(&c);       //此命令执行后父进程会有什么变化
9 Pthread_mutex_unlock(&m);
10 }
11
12 void *child(void *arg) {
13 printf("child\n");
14 thr_exit();
15 return NULL;
16 }

17
18 void thr_join() {
19 Pthread_mutex_lock(&m);  //为什么要有这行代码
20 while (done == 0)
21 Pthread_cond_wait(&c, &m);
22 Pthread_mutex_unlock(&m);
23 }
24 //父线程创建子线程后,子线程有可能直接运行结束,也可能不立刻运行
25 int main(int argc, char *argv[]) {
26 printf("parent: begin\n");
```

```
27 pthread_t p;
28 Pthread_create(&p, NULL, child, NULL);
29 thr_join();
30 printf("parent: end\n");
31 return 0;
    32 }
```

8. (5)

a) 右图为一个数组在虚拟内存中的地址，该虚拟内存用 4 位 VPN 和 4 位偏移量记录地址，并且对应一个页表，访问右图中数组时哪些元素需要查询页表

b) 提供了页表，查询这几个元素的物理地址(使用二进制表示)



9. 在#后写注释: (5)

a)

```
    def start(self):
        try:
            self.server_socket.bind((self.host, self.port)) #
            self.server_socket.listen(5) #
            print(f'Server started on {self.host}:{self.port}. Waiting
for connections...')

            while True:
                conn, addr = self.server_socket.accept() #
                client_thread =
                threading.Thread(target=self.handle_client, args=(conn,
                addr))
                client_thread.daemon = True
                client_thread.start()
        except Exception as e:
            print(f"Server error: {e}")
```

```
        finally:
            self.server_socket.close()
```

b)

```
    def start(self):
            try:
            self.client_socket.connect((self.host, self.port)) #
            self.running = True

            # Get username from user
            self.username = input("Enter your username: ")
            self.client_socket.sendall(self.username.encode('utf-8'))

            # Start receive thread
            receive_thread =
threading.Thread(target=self.receive_messages) #
            receive_thread.daemon = True
            receive_thread.start()

            print("Connected to server. Type 'exit' to quit.")
            self.send_messages()

        except Exception as e:
            print(f"Connection error: {e}")
        finally:
            self.client_socket.close()
            print("Disconnected from server.")
```

c)

```
    def send_messages(self):
        # Send messages to server from user input
        while self.running:
            try:
                message = input("You: ")
                if not self.running:
                    break
                if message.lower() == 'exit':
                    self.client_socket.sendall(message.encode('utf-8'))
                    self.running = False
                    break
                self.client_socket.sendall(message.encode('utf-8')) #
            except Exception as e:
                print(f"Error sending message: {e}")
                self.running = False
```

10. 在 Linux 系统中，假设 “home\student\st01”已创建，使用 open(“home\student\st01”, RDONLY)读取该文件，并读取三个数据块，请在下面的表格中对应位置用 read 或 write 标记对应的操作，时间顺序为自上向下(5)

	Data bitmap	Inode bitmap	Root inode	Student inode	St01 inode	Root data	St01 data	St01 [1]	St01[2]	St01[3]
Open ()										
Read( )										
Read( )										
Read( )										