目 录

25年计算机专升本专业课重要考点汇总

（全国通用版）

计算机专业考试（科目范围）

C语言程序设计、数据结构与算法、操作系统原理

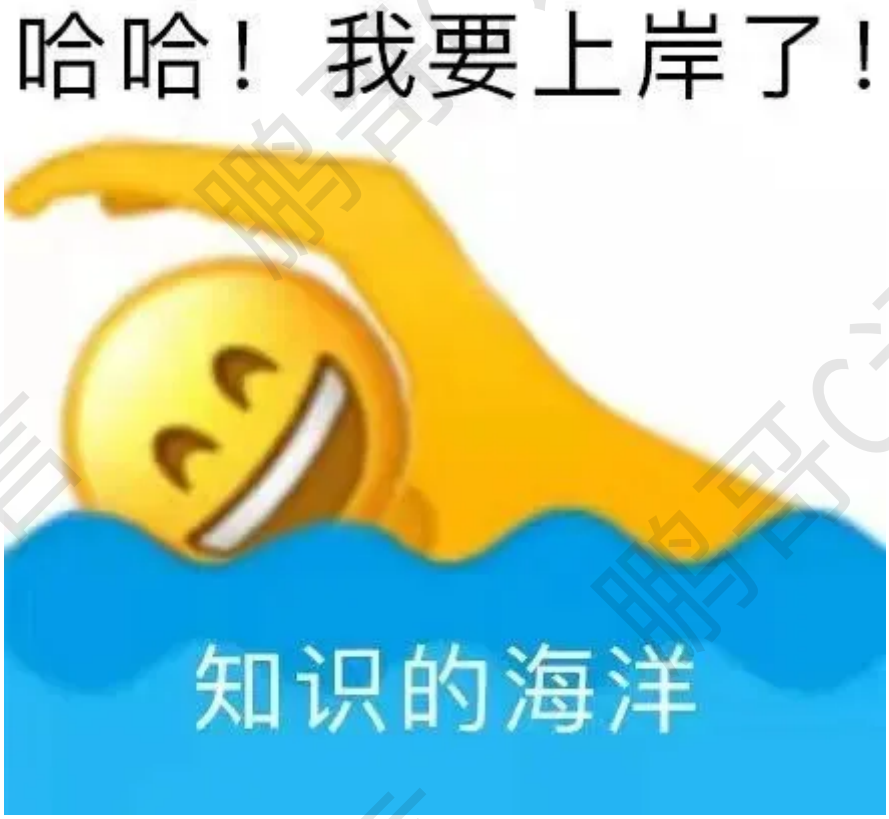
计算机网络、计算机组成原理、数据库系统、软件工程

考试题型

客观题（单选、多选、判断、填空）

程序设计题（程序阅读、程序填空、编程题）

应用题（大题提问）



本份资料包含十二个科目，覆盖各省常考科目。（均为高频考点提炼，根据自身备考科目学习即可）

12个科目名单

一、C语言程序设计 专升本高频考点背诵（极细化补充版）

二、数据结构与算法 专升本高频考点背诵（极细化补充版）

三、操作系统原理 专升本高频考点背诵（极细化补充版）

四、计算机网络 专升本高频考点背诵（极细化补充版）

五、计算机组成原理 专升本高频考点背诵（极细化补充版）

六、数据库系统 专升本高频考点背诵（极细化补充版）

七、软件工程 专升本高频考点背诵（极细化补充版）

八、计算机基础知识

九、Windows2000 系统使用

十、Word 2000 的使用

十一、Excel2000 的使用

十二、PowerPoint 的使用

你必上岸！

一、C语言程序设计 专升本高频考点背诵（极细化补充版）

（共171个高频考点+举例深入+考前速记）

C语言通识

1. C语言是一种**（结构化的程序设计语言）**，它强调**（模块化）**和**（层次化）**的程序设计方法，通过**（函数）**将程序分解为小的、易于管理的模块，每个模块完成特定的任务，从而提高代码的**（可读性）、（可维护性）**和**（可重用性）**。
2. C程序的执行总是从**（main函数）**开始，`main`函数是C程序的**（入口点）**，操作系统通过调用`main`函数来启动程序。`main`函数通常负责程序的**（初始化）、（调用其他函数）**以及**（返回程序的退出状态）**。
3. 每个C程序必须有且仅有一个**（main函数）**，`main`函数是程序的（唯一入口），如果缺少`main`函数，编译器将无法找到程序的起始位置。如果存在多个`main`函数，编译器将无法确定应该从哪个函数开始执行，从而导致**（编译错误）**。
4. C语言（区分大小写），C语言将**（大写字母）和（小写字母）**视为不同的字符。这意味着**（变量名）、（函数名）、（关键字）**等的大小写必须严格一致，否则编译器将无法识别。
5. C语言语句以**（分号;）**结束，分号是C语言语句的**（结束符）**，用于告诉编译器一条语句的结束位置。缺少分号会导致**（编译错误）**。一些语句，如**（预处理指令）和（复合语句）**，不需要以分号结尾。
6. 注释用于**（解释代码）**，`//`为**（单行注释）**，`/* */`为**（多行注释）**。注释是程序员用来解释代码的文本，编译器会忽略注释。良好的注释可以提高代码的**（可读性）和（可维护性）**。
7. 预处理指令以**（#）**开头，预处理指令是在**（编译之前）**由预处理器处理的指令。预处理器负责处理**（头文件包含）、（宏定义）、（条件编译）**等任务。
8. **（#include）用于（包含头文件）**，`#include`指令用于将头文件的内容包含到当前源文件中。头文件通常包含**（函数声明）、（宏定义）、（类型定义）**等信息，使得程序可以使用这些功能。
9. **（#define）用于（定义宏）**，`#define`指令用于定义宏，宏可以是**（对象宏）或（函数宏）**。对象宏用于定义常量，函数宏用于定义简单的函数。
10. 头文件通常包含**（函数声明）和（宏定义）**，头文件是包含函数声明、宏定义、类型定义等信息的文件，用于在多个源文件中**（共享这些信息）**。

数据类型和运算符

11. `char` 类型通常用于存储**（ASCII字符）**，`char` 类型占用（1个字节）的内存空间，可以存储范围**（-128到127）**之间的整数，或者**（0到255）**之间的无符号整数。
12. `int` 类型的大小取决于**（编译器）和（操作系统）**，`int` 类型用于存储整数，通常为**（4个字节）或（2个字节）**。可以使用**（sizeof(int)）**运算符获取 `int` 类型的大小。
13. `float` 类型提供**（单精度浮点数）**，`float` 类型用于存储带有小数部分的数值，通常占用**（4个字节）**的内存空间，可以表示大约**（6到7位）**有效数字。
14. `double` 类型提供**（双精度浮点数）**，**（精度更高）**，`double` 类型用于存储带有小数部分的数值，通常占用**（8个字节）**的内存空间，可以表示大约**（15到16位）**有效数字。
15. `long double` 类型提供**（扩展精度浮点数）**，`long double` 类型的大小和精度取决于**（编译器）和（操作系统）**，通常比 `double` 类型**（更大）、（精度更高）**。
16. 短整型 `short` 通常占用**（2个字节）**，`short` 类型用于存储整数，可以存储**（-32768到32767）**之间的整数，或者**（0到65535）**之间的无符号整数。
17. 长整型 `long` 通常占用**（4个字节）**，`long` 类型用于存储整数，可以存储**（-2147483648到2147483647）**之间的整数，或者**（0到4294967295）**之间的无符号整数。
18. 长长整型 `long long` 通常占用**（8个字节）**，`long long` 类型用于存储整数，可以存储**（非常大的整数）**。
19. 无符号整数类型使用**（unsigned）**关键字，`unsigned` 关键字用于修饰整数类型，表示**（无符号整数）**。
20. 无符号整数**（不能表示负数）**，无符号整数只能存储**（非负整数）**。
21. 可以使用**（十六进制：0x开头）或（八进制：0开头）**表示整数，例如，`0x1A` 表示十进制的26，`032` 表示十进制的26。
22. 浮点数可以使用**（科学计数法）**表示，例如，`1.23e4` 表示1.23乘以10的4次方，即12300。
23. 字符常量可以使用**（ASCII码）**表示，例如，`'\65'` 表示字符 `'A'`，因为 `'A'` 的ASCII码是65。
24. 字符串常量是**（字符数组）**，以（**空字符\0**）结尾，例如，`"Hello"` 实际上是一个包含 `'H'`，`'e'`，`'l'`，`'l'`，`'o'`，`'\0'` 的字符数组。
25. 字符串常量存储在**（只读内存区域）**，这意味着程序不能修改字符串常量的内容。

了解即可

1. 可以使用（`L` 前缀）表示宽字符常量，如 `L'A'`，宽字符类型是 `wchar_t`，用于表示（Unicode字符）。
2. 可以使用（`u` 前缀）表示UTF-16字符常量，如 `u'A'`（C11标准），UTF-16是一种Unicode编码方式。

3. 可以使用（`U`前缀）表示UTF-32字符常量，如`U'A'`（C11标准），UTF-32是另一种Unicode编码方式。
4. 可以使用（`R`前缀）表示原始字符串常量，避免转义字符，如`R"(Hello\nWorld)"`（C11标准），原始字符串常量中的`\n`不会被解释为换行符。

26. 算术运算符
    - 包含：+（加）、-（减）、*（乘）、/（除）、%（取余）、++（自增）、--（自减）
    - 特点：用于数值计算，如a=5/2结果为2（整数除法），%仅用于整数。

27. 赋值运算符
    - 基本形式：=（简单赋值），以及+=、-=、*=等复合赋值（如a+=5等价于a=a+5）
    - 特性：右结合性，例如a=b=c=5从右向左赋值。

28. 关系运算符
    - 包含：>（大于）、<（小于）、==（等于）、!=（不等于）、>=（大于等于）、<=（小于等于）
    - 作用：比较操作数大小，结果为逻辑值（0或1），如5>3返回1。

29. 逻辑运算符
    - 包含：&&（逻辑与）、||（逻辑或）、!（逻辑非）
    - 短路特性：&&左操作数为假时右操作数不执行，||左操作数为真时右操作数不执行。

30. 位运算符
    - 包含：&（按位与）、|（按位或）、^（按位异或）、~（按位取反）、<<（左移）、>>（右移）
    - 用途：直接操作二进制位，如a<<2表示将a的二进制左移两位。

31. 条件运算符（三目运算符）
    - 形式：表达式1 ? 表达式2 : 表达式3
    - 规则：若表达式1为真，返回表达式2的值，否则返回表达式3的值，如a>b ? a : b。

32. 其他特殊运算符
    - sizeof：计算变量或类型的字节数，如sizeof(int)结果为4。
    - 逗号运算符：连接多个表达式，最终结果为最后一个表达式值，如a=(x=3, y=5)时a=5。

按操作数数目分类

一元运算符

- **特点**：仅需一个操作数，如 `++`、`--`、`!`（逻辑非）、`~`（按位取反）。

二元运算符

- **特点**：需两个操作数，如 `+`、`-`、`&&`、`||` 等，占运算符的绝大多数。

三元运算符

- **唯一代表**：条件运算符 `?:`，如 `a>b ? a : b`。

33. 运算符优先级最高的是**（一元运算符）**，如 `++`、`--`、`!` 等，其次是**（算术运算符）**，最后是**赋值运算符**。
34. 自增/自减运算符 `++`、`--` 的前置与后置区别：**（前置先运算后赋值，后置先赋值后运算）**，例如 `a=3; b=++a;` 结果为 `a=4,b=4`。
35. 逻辑运算符 `&&` 和 `||` 具有**（短路特性）**，若左操作数已确定结果，则右操作数不执行。
36. 条件运算符 `? :` 是唯一的三目运算符，结合方向为**（右→左）**，例如 `a>b ? a : c>d ? c : d` 等价于 `a>b ? a : (c>d ? c : d)`。
37. 逗号运算符 `,` 优先级**（最低）**，整个表达式值为最后一个子表达式结果，例如 `a=(x=3,y=5)` 时 `a=5`。
38. 赋值运算符 `=` 结合方向为**（右→左）**，连续赋值时先计算右侧表达式，例如 `a=b=c=5` 等价于 `a=(b=(c=5))`。
39. 位运算符中，`<<` 左移右侧补0，`>>` 右移补符号位**（算术右移）**或补0**（逻辑右移，依赖编译器）**。
40. 运算符 `sizeof` 用于**（计算变量或类型所占字节数）**，是编译时运算符，例如 `sizeof(int)` 结果为4。
41. 关系运算符 `==` 与赋值运算符 `=` 易混淆，建议将常量写在左边避免误用，例如 `if(5==a)`。
42. 复合赋值运算符如 `+=`、`*=` 等价于展开式，但**（运算对象仅计算一次）**，例如 `a*=b+1` 等价于 `a=a*(b+1)`。

位运算

43. 位运算符用于**（直接操作整数在内存中存储的二进制位）**

44. `&` 运算符是**（按位与）**，它会将两个操作数的对应位进行比较，**只有当两个操作数对应位都为 1 时，结果的对应位才为 1，否则为 0。**

> 🖐 **例子：** 假设有两个整数 `a = 5` 和 `b = 3`，它们的二进制表示分别为 `a = 00000101` 和 `b = 00000011`（假设是 8 位整数）。
>
> - `a & b = 00000101 & 00000011 = 00000001`，转换为十进制为 1。

- **C代码示例：**

```c
int a = 5;  // 二进制：00000101
int b = 3;  // 二进制：00000011
int result = a & b; // result 的值为 1，二进制：00000001
printf("a & b = %d\n", result); // 输出：a & b = 1
```

45. `|` 运算符是**（按位或）**，它会将两个操作数的对应位进行比较，**只要两个操作数对应位中有一个为 1，结果的对应位就为 1，否则为 0。**

> 🖐 **例子：** 假设有两个整数 `a = 5` 和 `b = 3`，它们的二进制表示分别为 `a = 00000101` 和 `b = 00000011`（假设是 8 位整数）。
>
> - `a | b = 00000101 | 00000011 = 00000111`，转换为十进制为 7。

- **C代码示例：**

```c
int a = 5;  // 二进制：00000101
int b = 3;  // 二进制：00000011
int result = a | b; // result 的值为 7，二进制：00000111
printf("a | b = %d\n", result); // 输出：a | b = 7
```

46. `^` 运算符是**（按位异或）**，它会将两个操作数的对应位进行比较，**当两个操作数对应位不同时，结果的对应位为 1，相同时为 0。**

🖐 **例子：** 假设有两个整数 `a = 5` 和 `b = 3`，它们的二进制表示分别为 `a = 00000101` 和 `b = 00000011`（假设是 8 位整数）。

- `a ^ b = 00000101 ^ 00000011 = 00000110`，转换为十进制为 6。

- **C代码示例：**

```c
int a = 5;  // 二进制：00000101
int b = 3;  // 二进制：00000011
int result = a ^ b; // result 的值为 6，二进制：00000110
printf("a ^ b = %d\n", result); // 输出：a ^ b = 6
```

47. `~` 运算符是**（按位取反）**，它会**将操作数的每一位进行取反，即 0 变为 1，1 变为 0。**

注意： 在有符号整数中，取反会影响符号位，需要考虑补码表示。

🖐 **例子：** 假设有一个 8 位整数 `a = 5`，它的二进制表示为 `a = 00000101`。

- `~a = ~00000101 = 11111010`，转换为十进制为 -6（补码表示）。

- **C代码示例：**

```c
int a = 5;  // 二进制：00000101 (8位)
int result = ~a; // result 的值为 -6，二进制：11111010 (补码)
printf("~a = %d\n", result); // 输出：~a = -6
```

48. `<<` 运算符是**（左移）**，它会**将操作数的二进制位向左移动指定的位数，右边补 0。左移一位相当于乘以 2。**

🖐 **例子：** 假设有一个整数 `a = 5`，它的二进制表示为 `a = 00000101`。

- `a << 2 = 00000101 << 2 = 00010100`，转换为十进制为 20。

- **C代码示例：**

```c
int a = 5;  // 二进制：00000101
```

```
int result = a << 2; // result 的值为 20，二进制：00010100
printf("a << 2 = %d\n", result); // 输出：a << 2 = 20
```

49. `>>` 运算符是（**右移**），它会**将操作数的二进制位向右移动指定的位数**。右移分为两种：**逻辑右移**（左边补 0）和 **算术右移**（左边补符号位）。具体使用哪种右移方式取决于编译器和数据类型。

> 🖐 **例子：**
>
> 假设有一个整数 `a = 5`，它的二进制表示为 `a = 00000101`。
>
> - `a >> 2 = 00000101 >> 2 = 00000001`，转换为十进制为 1（逻辑右移）。
>
> 假设有一个整数 `a = -5`，它的二进制表示为 `a = 11111011`（补码）。
>
> - `a >> 2 = 11111011 >> 2 = 11111110`，转换为十进制为 -2（算术右移，补符号位）。

- **C代码示例：**

```
int a = 5;  // 二进制：00000101
int result = a >> 2; // result 的值为 1，二进制：00000001
printf("a >> 2 = %d\n", result); // 输出：a >> 2 = 1
int b = -5; // 二进制：11111011（补码）
int result2 = b >> 2; // result2 的值取决于编译器，可能是 -2 或其他值
printf("b >> 2 = %d\n", result2); // 输出：b >> 2 = -2（取决于编译器）
```

50. （位设置）用于（将一个整数的特定位设置为 1）。

> 🖐 **例子：** 假设要将一个 8 位整数 `num = 0b00000000` 的第 2 位（从右往左数，从 0 开始）设置为 1。
>
> - 定义一个掩码 `mask = 0b00000100`。
> - `newNum = num | mask = 0b00000000 | 0b00000100 = 0b00000100`，即十进制的 4。

- **C代码示例：**

```c
unsigned char num = 0b00000000; // 初始值为 0
unsigned char mask = 0b00000100; // 掩码
unsigned char newNum = num | mask; // 设置第 2 位为 1
printf("New number: %d\n", newNum); // 输出：New number: 4
```

51. （位清除）用于（将一个整数的特定位设置为 0）。

> 🖐 **例子：** 假设要将一个 8 位整数 `num = 0b11111111` 的第 2 位（从右往左数，从 0 开始）设置为 0。
>
> - 定义一个掩码 `mask = ~0b00000100 = 0b11111011`。
> - `newNum = num & mask = 0b11111111 & 0b11111011 = 0b11111011`，即十进制的 251。

- **C代码示例：**

```c
unsigned char num = 0b11111111; // 初始值为 255
unsigned char mask = ~0b00000100; // 掩码
unsigned char newNum = num & mask; // 清除第 2 位
printf("New number: %d\n", newNum); // 输出：New number: 251
```

52. （判断奇偶数）可以使用位运算来（快速判断一个整数是奇数还是偶数）。

> 🖐 **例子：**
>
> - 如果 `num = 7`，则 `num & 1 = 0b00000111 & 0b00000001 = 0b00000001 = 1`，所以 7 是奇数。
> - 如果 `num = 6`，则 `num & 1 = 0b00000110 & 0b00000001 = 0b00000000 = 0`，所以 6 是偶数。

- **C代码示例：**

```c
int num = 7;
if (num & 1) {
    printf("%d 是奇数\n", num); // 输出：7 是奇数
} else {
```

```
    printf("%d 是偶数\n", num);
}

int num2 = 6;
if (num2 & 1) {
    printf("%d 是奇数\n", num2);
} else {
    printf("%d 是偶数\n", num2); // 输出: 6 是偶数
}
```

53. （交换两个变量的值）可以使用位运算来（在不使用额外变量的情况下交换两个变量的值）。

🖐 **例子：** 假设 `a = 5` 和 `b = 3`。

- `a ^= b;` // `a = 5 ^ 3 = 6` (0101 ^ 0011 = 0110)
- `b ^= a;` // `b = 3 ^ 6 = 5` (0011 ^ 0110 = 0101)
- `a ^= b;` // `a = 6 ^ 5 = 3` (0110 ^ 0101 = 0011)

- **C代码示例：**

```
int a = 5;
int b = 3;
printf("交换前: a = %d, b = %d\n", a, b); // 输出: 交换前: a = 5, b = 3

a ^= b;
b ^= a;
a ^= b;

printf("交换后: a = %d, b = %d\n", a, b); // 输出: 交换后: a = 3, b = 5
```

进制转换

54. 进制是一种**（计数系统）**，用于表示数值。常见的进制包括**（二进制）、（八进制）、（十进制）和（十六进制）**。

55. **（二进制）**使用**0 和 1**两个数字表示数值，是计算机内部使用的基本进制。

56. **（八进制）**使用**0 到 7**八个数字表示数值，通常用于简化二进制的表示。

57. **（十进制）**使用**0 到 9**十个数字表示数值，是人类最常用的进制。

58. **（十六进制）**使用 **0 到 9** 和 **A 到 F** 十六个数字表示数值，通常用于表示内存地址和颜色值。
59. 在 C 语言中，可以使用 **（不同的前缀）**来表示不同进制的整数常量。
60. **（十进制常量）**直接使用**数字**表示，例如 `123` 。
61. **（八进制常量）**以 `0` 开头，例如 `0173` 表示十进制的 123。
62. **（十六进制常量）**以 **0x 或 0X** 开头，例如 `0x7B` 表示十进制的 123。
63. C 语言 **（没有直接表示二进制常量的语法）**，但可以使用位运算或查表法来实现二进制的转换。

进制转换方法

64. **（十进制转换为二进制）**可以使用 **（除 2 取余法）**，**将十进制数不断除以 2，记录每次的余数，直到商为 0，然后将余数倒序排列。**

> 🖐 **例子：** 将十进制数 25 转换为二进制：
>
> - 25 / 2 = 12 ... 1
> - 12 / 2 = 6 ... 0
> - 6 / 2 = 3 ... 0
> - 3 / 2 = 1 ... 1
> - 1 / 2 = 0 ... 1
> - 所以，25 的二进制表示为 11001。

65. **（二进制转换为十进制）**可以**将二进制数的每一位乘以 2 的相应次方，然后将结果相加。**

> 🖐 **例子：** 将二进制数 11001 转换为十进制：
>
> - 1 * 2^4 + 1 * 2^3 + 0 * 2^2 + 0 * 2^1 + 1 * 2^0 = 16 + 8 + 0 + 0 + 1 = 25
> - 所以，11001 的十进制表示为 25。

66. **（十进制转换为八进制）**可以使用 **（除 8 取余法）**，**将十进制数不断除以 8，记录每次的余数，直到商为 0，然后将余数倒序排列。**

> 🖐 **例子：** 将十进制数 123 转换为八进制：
>
> - 123 / 8 = 15 ... 3

- 15 / 8 = 1 ... 7
- 1 / 8 = 0 ... 1
- 所以，123 的八进制表示为 173。

67. **（八进制转换为十进制）**可以**将八进制数的每一位乘以 8 的相应次方，然后将结果相加。**

🖐 **例子：** 将八进制数 173 转换为十进制：

- 1 * 8^2 + 7 * 8^1 + 3 * 8^0 = 64 + 56 + 3 = 123
- 所以，173 的十进制表示为 123。

68. **（十进制转换为十六进制）**可以使用**（除 16 取余法），将十进制数不断除以 16，记录每次的余数，如果余数大于 9，则用 A 到 F 表示，直到商为 0，然后将余数倒序排列。**

🖐 **例子：** 将十进制数 123 转换为十六进制：

- 123 / 16 = 7 ... 11 (B)
- 7 / 16 = 0 ... 7
- 所以，123 的十六进制表示为 7B。

69. **（十六进制转换为十进制）**可以**将十六进制数的每一位乘以 16 的相应次方，然后将结果相加。**

🖐 **例子：** 将十六进制数 7B 转换为十进制：

- 7 * 16^1 + 11 * 16^0 = 112 + 11 = 123
- 所以，7B 的十进制表示为 123。

70. **（二进制转换为八进制）**可以**将二进制数从右向左每 3 位一组进行分组，然后将每组转换为对应的八进制数。**

🖐 **例子：** 将二进制数 11001 转换为八进制：

- 011 001 (从右向左分组，不足 3 位用 0 补齐)

- 3 1
- 所以，11001 的八进制表示为 31。

71. **（八进制转换为二进制）**可以**将八进制数的每一位转换为对应的 3 位二进制数。**

🖐 **例子：** 将八进制数 31 转换为二进制：

- 3 -> 011
- 1 -> 001
- 所以，31 的二进制表示为 011001，去掉前面的 0，为 11001。

72. **（二进制转换为十六进制）**可以**将二进制数从右向左每 4 位一组进行分组，然后将每组转换为对应的十六进制数。**

🖐 **例子：** 将二进制数 11001 转换为十六进制：

- 0001 1001 (从右向左分组，不足 4 位用 0 补齐)
- 1 9
- 所以，11001 的十六进制表示为 19。

73. **（十六进制转换为二进制）**可以**将十六进制数的每一位转换为对应的 4 位二进制数。**

🖐 **例子：** 将十六进制数 19 转换为二进制：

- 1 -> 0001
- 9 -> 1001
- 所以，19 的二进制表示为 00011001，去掉前面的 0，为 11001。

74. **源码、反码、补码**

1. **源码（Original Code）：**
   源码是最简单的机器数表示形式，直接将十进制数转换为二进制数，并在最左边添加符号位。
   符号位：0 表示正数，1 表示负数。
   例如，十进制数 +7 的源码为 00000111（假设是 8 位），-7 的源码为 10000111。
2. **反码（Inverse Code）：**
   正数的反码与源码相同。
   负数的反码是对其源码除符号位外的所有位取反（0 变为 1，1 变为 0）。
   例如，+7 的反码为 00000111，-7 的反码为 11111000。
3. **补码（Complement Code）：**
   正数的补码与源码相同。
   负数的补码是其反码加 1。
   例如，+7 的补码为 00000111，-7 的补码为 11111001。
4. **总结**

十进制	源码	反码	补码
7	111	111	111
-7	10000111	11111000	11111001

正数的原反补码一致。

75. **原反补转换过程例子**

- **例子 1：十进制数 +10**
  源码： 00001010 (8 位表示)
  反码： 00001010 (正数，与源码相同)
  补码： 00001010 (正数，与源码相同)

- **例子 2：十进制数 -10**
  源码： 10001010 (8 位表示)
  反码： 11110101 (源码符号位不变，其余位取反)
  补码： 11110110 (反码加 1)

- **例子 3：已知补码求源码**
  如果已知一个负数的补码，求其源码，可以对补码再次求补码（即取反加 1），或者 补码-1，再取反。
  假设已知某数的补码为 11110110。

a. 判断符号位为 1，是负数。

b. 反码：11110110 - 1 = 11110101

c. 源码：符号位不变，对反码取反 = 10001010，即 -10。

控制流

76. `if` 语句可以**（嵌套）**，`if` 语句可以嵌套在另一个 `if` 语句内部，形成**（嵌套的条件判断结构）**，用于处理更复杂的条件判断逻辑。

77. `else if` 语句用于处理**（多个互斥条件）**，`else if` 语句是 `if` 语句的**（扩展）**，可以提供**（多个备选的条件判断分支）**。

78. `switch` 语句的 `case` 标签必须是**（常量表达式）**，`case` 标签的值必须是**（唯一的）**，不能重复。（标签必须是整形）

79. `switch` 语句可以**（没有default标签）**，如果 `switch` 语句没有 `default` 标签，并且没有一个 `case` 标签的值与 `switch` 表达式的值匹配，那么 `switch` 语句将**（不执行任何操作）**。

80. `while` 循环的条件在**（每次迭代前检查）**，如果条件为真，则执行**（循环体）**；否则，跳出循环。`while` 循环可能**（一次也不执行循环体）**。

81. `do...while` 循环**（至少执行一次代码块）**，`do...while` 循环与 `while` 循环类似，但是 `do...while` 循环的条件在**（每次迭代后检查）**。

82. `for` 循环的初始化、条件和更新表达式可以**（省略）**，如果省略了初始化表达式，那么循环**（不会进行初始化操作）**；如果省略了条件表达式，那么循环将**（无限循环）**。

83. 可以使用 `break` 语句跳出**（多层嵌套循环）**，`break` 语句只能跳出**（当前循环）**，不能直接跳出多层嵌套循环。

84. 可以使用 `continue` 语句跳过**（当前循环迭代的剩余部分）**，`continue` 语句**（不会跳出循环）**。

85. `goto` 语句可以跳转到**（同一函数内的任何标签位置）**，`goto` 语句应该**（谨慎使用）**，因为它可能导致代码**（难以理解和维护）**。

函数

86. 函数可以**（没有参数）**，表示该函数不需要接收任何输入数据，通常用于执行一些**（独立的操作）**。

87. 函数可以**（有多个参数）**，用于接收多个输入数据，参数之间用**（逗号,）**分隔，参数的类型和顺序必须与函数声明中的参数类型和顺序**（一致）**。

88. 函数参数可以是**（任何数据类型）**，包括基本数据类型、结构体、共用体、指针等，参数的类型决定了函数可以接收的**（数据种类）**。

89. 函数参数传递可以是**（值传递）或（地址传递）**，值传递是指将参数的值复制给函数，地址传递是指将参数的地址传递给函数。

90. 值传递是指将**（参数的值复制给函数的形式参数）**，在函数内部对形式参数的修改**（不会影响实际参数的值）**，可以保护实际参数的值不被修改。

91. 地址传递是指将**（参数的地址传递给函数的形式参数）**，在函数内部对形式参数指向的内存空间进行修改**（会影响实际参数的值）**，可以用于在函数内部修改实际参数的值。

92. 在函数内部修改**（值传递的参数）**不会影响原始变量，因为函数操作的是**（参数的副本）**。

93. 在函数内部修改**（地址传递的参数）**会影响原始变量，因为函数操作的是**（原始变量的内存地址）**。

94. 函数可以返回**（任何数据类型）**，包括基本数据类型、结构体、共用体、指针等，返回值的类型决定了函数可以返回的（数据种类）。

95. 函数可以使用**（return语句）**返回值，`return`语句用于将函数的结果返回给**（调用者）**，可以出现在函数的**（任何位置）**。

数组和指针

96. 数组元素在内存中是**（连续存储的）**，这意味着可以通过**（指针运算）**来访问数组元素。

97. 数组名是指向**（数组第一个元素的指针）**，这意味着可以使用数组名来**（访问数组元素）**。

98. 可以使用**（指针）**访问数组元素，通过将指针指向数组的某个元素，然后使用指针运算来访问其他元素。

99. **`*(arr + i)`等价于（`arr[i]`）**，都表示访问数组`arr`的第`i`个元素，前者是指针运算，后者是数组下标运算。

100. 指针可以进行**（加减运算）**，用于在内存中移动指针的位置，指针加1表示指向**（下一个相同类型的元素）**。

101. 指针加1表示指向**（下一个相同类型的元素）**，例如，如果`p`是一个指向`int`类型变量的指针，那么`p + 1`将指向下一个`int`类型变量的内存地址。（移动sizeof（类型）个字节）

102. 指针减1表示指向**（上一个相同类型的元素）**，例如，如果`p`是一个指向`int`类型变量的指针，那么`p - 1`将指向上一个`int`类型变量的内存地址。

103. 可以使用**（指针）**遍历数组，通过将指针指向数组的第一个元素，然后使用指针运算来依次访问数组的每个元素。

104. 可以使用**（指针作为函数参数）**传递数组，通过将数组名作为参数传递给函数，函数可以访问和修改数组的元素。
105. 在函数内部修改**（指针参数指向的数组元素）**会影响原始数组，因为指针传递的是数组的**（地址）**。
106. 多维数组是**（数组的数组）**，例如，二维数组可以看作是**（行和列组成的表格）**。
107. 多维数组元素在内存中也是**（连续存储的）**，但其访问方式需要考虑**（行和列的索引）**。
108. 字符串是**（字符数组）**，以**（空字符 `\0`）**结尾，空字符用于**（标记字符串的结束）**。
109. 字符串常量存储在**（只读内存区域）**，程序**（不能修改）**字符串常量的内容。
110. 可以使用**（strlen函数）**获取字符串的长度，`strlen` 函数长度计数不包括**（空字符 `\0`）**。
111. 可以使用**（strcpy函数）**复制字符串，`strcpy` 函数会将源字符串的**（空字符\0）**也复制到目标字符串。
112. 可以使用**（strcat函数）**连接字符串，`strcat` 函数会将源字符串连接到目标字符串的**（末尾）**，并添加**（空字符\0）**。
113. 可以使用**（strcmp函数）**比较字符串，`strcmp` 函数**返回（0）表示两个字符串相等，返回（正数）表示第一个字符串大于第二个字符串，返回（负数）表示第一个字符串小于第二个字符串。**
114. 可以使用**（strstr函数）**在一个字符串中查找子字符串，`strstr` 函数返回子字符串在字符串中（第一次出现的位置的指针），返回（`NULL`）表示子字符串不存在。
115. 指针是一种**（特殊的变量）**，用于**（存储内存地址）**。
116. 指针可以指向**（任何数据类型）**，包括**（基本数据类型）、（结构体）、（联合体）、（函数）**等。
117. 可以使用**（* 运算符）**来**（解引用指针）**，获取指针指向的内存地址中存储的值。
118. 可以使用**（& 运算符）**来（获取变量的地址）。
119. 指针可以进行**（加减运算）**，但需要注意**（指针类型）**，指针加 1 实际上增加的是**（一个数据类型的大小）**。
120. 可以使用**（函数指针）**来**（指向函数）**，实现**（回调函数）和（动态函数调用）**。

结构体和共用体

121. 结构体是**（不同类型成员的集合）**，是一种用户自定义的数据类型，可以将不同类型的成员组合在一起，用于表示**（复杂的数据结构）**。
122. 结构体成员在内存中是**（连续存储的）**，这意味着可以使用**（指针运算）**来访问结构体成员。

123. 结构体成员可以通过**（.运算符）**访问，`.`运算符用于访问结构体变量的成员，例如，`student.name`表示访问`student`结构体变量的`name`成员。
124. 结构体指针可以通过**（->运算符）**访问成员，`->`运算符用于访问结构体指针指向的结构体变量的成员，例如，`studentPtr->name`表示访问`studentPtr`指针指向的结构体变量的`name`成员。
125. 结构体可以**（嵌套）**，结构体可以嵌套在另一个结构体内部，形成嵌套的结构体，用于表示**（更复杂的数据结构）**。
126. 可以使用**（typedef关键字）**为结构体定义别名，这可以简化结构体类型的使用，例如，`typedef struct { int x, y; } Point;`。
127. 可以使用**（位域）**在结构体中定义占用**（特定位数的成员）**，位域可以**（节省内存空间）**，但使用时需要注意**（可移植性）**。
128. 共用体是**（不同类型成员共享同一内存位置）**的数据结构，这意味着在任何给定时间，共用体只能存储**（一个成员的值）**。
129. 枚举类型使用**（enum关键字）**定义，用于定义一组**（命名的整数常量）**，可以提高代码的**（可读性）**。
130. 可以使用**（sizeof运算符）**获取结构体或共用体的大小，结构体的大小可能受到**（内存对齐）**的影响。
131. 内存对齐是指**将数据存储在内存地址是其大小的倍数的位置**，这可以提高程序的（性能）。
132. 结构体可以包含**（指向自身的指针）**，这允许创建**（链表）**、**（树）**等（递归数据结构）。
133. 结构体可以包含**（函数指针）**，这允许在结构体中存储（函数），实现（多态性）。
134. 可以使用**（柔性数组）**作为结构体的最后一个成员，柔性数组的大小可以在（运行时动态确定）。
135. 结构体和联合体的大小受到（内存对齐）的影响，可以使用`#pragma pack`控制对齐方式。

内存管理

136. **（malloc函数）**用于**（动态分配内存）**，`malloc`函数在**（堆）**上动态分配内存，接受一个参数，表示需要分配的内存大小**（以字节为单位）**。
137. `malloc`函数返回**（分配的内存块的指针）**，如果内存分配成功，`malloc`函数返回一个指向分配的内存块的指针。
138. `malloc`函数**返回（NULL）表示（内存分配失败）**，内存分配失败可能是因为**（系统内存不足）**。

139. （`calloc`函数）用于**（分配并初始化内存）**，`calloc`函数在**（堆）**上分配内存，并将分配的内存块**（初始化为0）**，接受两个参数，第一个参数表示需要分配的元素个数，第二个参数表示每个元素的大小（以字节为单位）。

140. `calloc`函数将**（分配的内存块数据设置为0）**，这意味着分配的内存块中的每个字节都被设置为0。

141. `malloc`函数返回**（void*类型）**的指针，需要（强制类型转换）为所需的类型。

142. `realloc`函数用于**（改变已分配内存块的大小）**，可以扩大或缩小内存块。

143. `free`函数用于**（释放动态分配的内存）**，释放内存后，指针应该设置为**（NULL）**，以避免**（悬挂指针）**。

144. 内存泄漏是指**（分配的内存没有被释放）**，长时间运行的程序中的内存泄漏会导致**（系统资源耗尽）**。

145. 悬挂指针是指**（指向已释放内存的指针）**，访问悬挂指针会导致（未定义行为）。

文件操作

146. 文件是**（存储在磁盘上的数据集合）**，可以是文本文件、二进制文件等，是操作系统管理数据的**（基本单位）**。

147. 文件操作需要**（先打开文件）**，打开文件可以建立程序与文件之间的连接，使得程序可以读取或写入文件的数据。

148. （`fopen`函数）用于**（打开文件）**，`fopen`函数接受两个参数，第一个参数是（文件名），第二个参数是（文件打开模式）。

149. `fopen`函数返回**（文件指针）**，文件指针是指向**（文件结构的指针）**，文件结构包含了文件的各种信息，例如文件名、文件打开模式、文件指针位置等。

150. `fopen`函数返回**（NULL）表示（文件打开失败）**，文件打开失败可能是因为**（文件不存在）、（文件权限不足）**等。

151. 文件打开模式包括（`"r"`、`"w"`、`"a"`、`"r+"`、`"w+"`、`"a+"`），这些模式决定了文件如何被访问。

152. `"r"`模式用于（只读方式打开文件），文件必须（存在）。

153. `"w"`模式用于（只写方式打开文件），如果文件存在则（覆盖），如果文件不存在则（创建）。

154. `"a"`模式用于（追加方式打开文件），如果文件不存在则（创建），写入的数据会添加到文件（末尾）。

155. `"r+"`模式用于（读写方式打开文件），文件必须（存在），可以读取和修改文件内容。

156. `"w+"` 模式用于（读写方式打开文件），如果文件存在则（覆盖），如果文件不存在则（创建），可以读取和修改文件内容。

157. `"a+"` 模式用于（读写方式打开文件），如果文件不存在则（创建），写入的数据会添加到文件（末尾），可以读取文件内容。

158. 可以使用（`"b"` 模式）以（二进制方式）打开文件，二进制方式不进行（文本转换）。

159. 可以使用（`"t"` 模式）以（文本方式）打开文件（默认），文本方式会进行（文本转换），例如，将换行符转换为特定的字符。

160. `fclose` 函数用于（关闭文件），关闭文件可以释放（文件资源）。

预处理

161. 对象宏用于定义**（常量）**，对象宏只是简单的**（文本替换）**，在编译之前进行。

162. `#undef` 用于**（取消宏定义）**，如果需要取消之前定义的宏，可以使用 `#undef` 指令。这可以避免宏名冲突或在特定代码段禁用宏。

163. 条件编译指令包括（**`#ifdef`**、**`#ifndef`**、**`#else`**、**`#endif`**），这些指令允许根据（条件）选择性地编译代码，常用于处理平台差异、调试开关等。

164. `#ifdef` 用于**（判断宏是否已定义）**，如果指定的宏已经定义，则编译 `#ifdef` 和 `#else` 之间的代码；否则，编译 `#else` 和 `#endif` 之间的代码。

165. `#ifndef` 用于**（判断宏是否未定义）**，如果指定的宏尚未定义，则编译 `#ifndef` 和 `#else` 之间的代码；否则，编译 `#else` 和 `#endif` 之间的代码。

166. `#else` 用于**（提供备选代码块）**，`#else` 指令必须与 `#ifdef` 或 `#ifndef` 指令配对使用，用于在条件不满足时编译另一段代码。

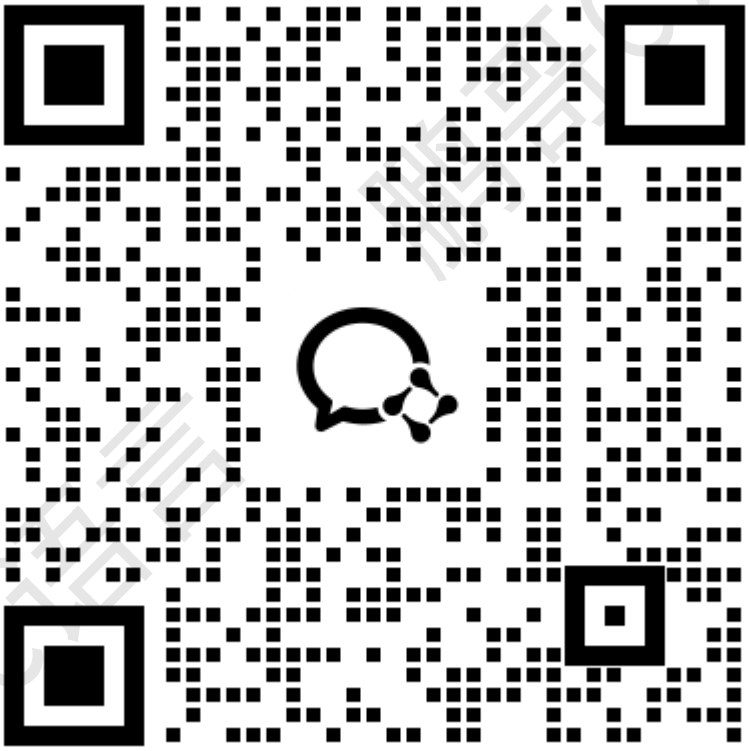
167. `#endif` 用于**（结束条件编译块）**，每个 `#ifdef`、`#ifndef`、`#if` 指令都必须以 `#endif` 指令结束。

168. `#if` 指令允许使用**（常量表达式）**进行条件编译，只有当常量表达式的值为非零时，才编译 `#if` 和 `#else` 之间的代码。

169. `#elif` 指令是 **`#else` 和 `#if` 的组合**，允许在多个条件之间进行选择，提供（多个备选代码块）。

170. `#line` 指令用于**（改变行号和文件名）**，这主要用于（调试）目的，可以使编译器报告的错误信息指向不同的位置。

171. `#error` 指令用于（生成编译错误），当编译器遇到 `#error` 指令时，会产生一个错误信息并终止编译，常用于（检查编译时错误）。



💡 需要更多C语言资料：如编译器、安装教程、C语言配套视频及刷题资料，扫描下方二维码添加助教领取

二、数据结构与算法 专升本高频考点背诵（极细化补充版）

（共131个高频考点+考前速记）

通识

1. 数据结构是研究**（非数值计算的程序设计问题中，计算机的操作对象）**以及它们之间的**（关系和运算）**等的学科，旨在提高程序效率和降低资源消耗。
2. 数据结构包括**（数据的逻辑结构、存储结构以及定义在数据上的运算）**三个密切相关的方面，缺一不可，共同决定了数据的组织方式和处理效率。
3. **（逻辑结构）**是从逻辑关系上描述数据，它与数据的**（具体存储方式无关）**，是独立于计算机的，包括**（线性结构、树形结构、图形结构和集合结构）**。
4. **（存储结构）**是数据在计算机中的表示**（也称映像）**，也称**（物理结构）**，它包括**（数据元素的表示）和（数据元素之间关系的表示）**，常见的存储结构有**（顺序存储、链式存储、索引存储和散列存储）**。
5. 数据类型是一个**（值的集合）**和定义在这个值集上的**（一组操作）**的总称，例如整型、浮点型和字符型等，不同的数据类型决定了数据的存储空间和操作方式。

6. 抽象数据类型**（ADT）**是指**（一个数学模型）**以及定义在该模型上的一组操作，它强调**（数据类型的抽象特性）和（操作的逻辑定义）**，不涉及具体的实现细节。
7. 算法是指**（解决特定问题求解步骤的描述）**，在计算机中表现为（指令的有限序列），是计算机解决问题的具体方法和步骤，直接影响程序的执行效率。
8. 算法的五个重要特性是：（有穷性，算法必须在执行有限步骤后结束）、（确定性，算法的每个步骤必须有确切的含义，无二义性）、（可行性，算法的每个步骤都必须能有效执行）、（输入，算法有零个或多个输入）、（输出，算法有一个或多个输出）。
9. 算法设计的要求包括：**（正确性，算法能够正确地解决问题）、（可读性，算法易于理解和修改）、（健壮性，算法能够处理各种异常情况）、（效率与低存储量需求，算法尽可能节省时间和空间资源）**。
10. 算法分析主要分析算法的**（时间复杂度和空间复杂度）**，旨在评估算法的效率和资源消耗，为算法选择和优化提供依据。
11. 算法的时间复杂度是指**（算法执行时间随输入规模增长的量级）**，常用大O表示法表示，例如O(1)、O(log n)、O(n)、O(n log n)、O(n^2)等，反映了算法执行效率的增长趋势。
12. 算法的空间复杂度是指**（算法执行过程中所需要的存储空间）**，包括**（算法本身所占空间、输入数据所占空间和辅助变量所占空间）**，空间复杂度越低，算法对内存资源的消耗越小。

线性表

13. 线性表是**（由n(n>=0)个数据元素组成的有限序列）**，数据元素类型相同，元素之间存在**（线性关系）**，是最基本、最常用的数据结构之一。
14. 线性表的逻辑特征是**（数据元素之间存在一对一的线性关系）**，即每个元素都有唯一的前驱和后继**（除了第一个元素没有前驱，最后一个元素没有后继）**，体现了线性表的有序性。
15. 线性表的存储结构主要有**（顺序存储结构）和（链式存储结构）**两种，不同的存储结构决定了线性表的操作效率和适用场景。
16. 顺序表是**（用一组地址连续的存储单元依次存储线性表的数据元素）**，通过**（数组）**来实现，可以随机访问任意位置的元素，但插入和删除操作效率较低。
17. 顺序表的优点是**（可以随机存取表中任一元素，查找速度快）**，缺点是**（插入和删除操作需要移动大量元素，容易造成存储空间浪费，表长固定）**。
18. 链表是**（用任意的存储单元来存储线性表的数据元素）**，这些存储单元可以连续，也可以不连续，通过**（指针）**来建立元素之间的逻辑关系，插入和删除操作效率较高，但不能随机访问。
19. 链表中每个结点包含**（数据域）和（指针域）**两部分，数据域存储数据元素本身，指针域存储指向下一个结点的指针，是链表实现的关键。
20. 链表的优点是**（插入和删除操作不需要移动元素，表长可变，存储空间利用率高）**，缺点是**（不能随机存取，查找效率较低，需要额外的存储空间存储指针）**。

21. 单链表是**（每个结点只有一个指针域的链表）**，指针域指向后继结点，只能单向遍历，实现简单，但查找前驱结点需要从头开始。
22. 双链表是**（每个结点有两个指针域的链表）**，分别指向前驱结点和后继结点，可以双向遍历，查找前驱结点效率高，但需要额外的存储空间维护前驱指针。
23. 循环链表是**（链表的最后一个结点的指针域指向头结点）**，形成一个环，从任意结点出发都可以遍历整个链表，适用于需要循环遍历的场景。
24. 顺序表适合**（元素个数变化不大，主要操作是查找）**的应用场景，例如静态数据的存储和查询，对存储空间要求不高，但需要频繁查找元素。
25. 链表适合**（元素个数变化大，主要操作是插入和删除）**的应用场景，例如动态数据的维护和更新，对存储空间利用率要求高，但需要频繁插入和删除元素。
26. 线性表的插入操作需要考虑**（插入位置的合法性，防止数组越界或指针错误）**，顺序表插入需要**（移动插入位置之后的所有元素，时间复杂度为O(n)）**，链表插入需要**（修改插入位置前后结点的指针，时间复杂度为O(1)，但需要先找到插入位置）**。
27. 线性表的删除操作需要考虑**（删除位置的合法性，防止数组越界或指针错误）**，顺序表删除需要**（移动删除位置之后的所有元素，时间复杂度为O(n)）**，链表删除需要**（修改删除位置前后结点的指针，时间复杂度为O(1)，但需要先找到删除位置）**。

栈和队列

28. 栈是**（只允许在一端进行插入或删除操作的线性表）**，该端称为**（栈顶）**，另一端称为**（栈底）**，是一种特殊的线性表，限制了插入和删除的位置。
29. 栈的特点是**（后进先出LIFO）**，即最后插入的元素最先被删除，类似于堆叠物品，后放的先取。
30. 栈的基本操作包括**（入栈push，将元素压入栈顶）和（出栈pop，将栈顶元素弹出）**，是栈的核心操作，用于实现数据的存储和访问。
31. 入栈操作是指**（在栈顶插入一个新元素）**，需要先判断栈是否已满，如果已满则发生**（栈溢出）**，否则将新元素放入栈顶，并修改栈顶指针。
32. 出栈操作是指**（删除栈顶元素）**，需要先判断栈是否为空，如果为空则发生**（栈下溢）**，否则将栈顶元素取出，并修改栈顶指针。
33. 栈的应用包括**（表达式求值，将中缀表达式转换为后缀表达式并计算结果）**、**（递归调用，保存函数调用的现场信息）**、**（括号匹配，检查括号是否成对出现）和（深度优先搜索，用于图的遍历）**等。
34. 队列是**（只允许在表的一端进行插入，另一端进行删除操作的线性表）**，插入端称为**（队尾）**，删除端称为**（队头）**，也是一种特殊的线性表，限制了插入和删除的位置。
35. 队列的特点是**（先进先出FIFO）**，即最先插入的元素最先被删除，类似于排队，先到先得。

36. 队列的基本操作包括**（入队enqueue，将元素插入队尾）**和**（出队dequeue，将队头元素删除）**，是队列的核心操作，用于实现数据的存储和访问。
37. 入队操作是指**（在队尾插入一个新元素）**，需要先判断队列是否已满，如果已满则发生**（队溢出）**，否则将新元素放入队尾，并修改队尾指针。
38. 出队操作是指**（删除队头元素）**，需要先判断队列是否为空，如果为空则发生**（队空）**，否则将队头元素取出，并修改队头指针。
39. 队列的应用包括**（广度优先搜索，用于图的遍历）**、**（任务调度，按照优先级顺序执行任务）**和**（消息队列，用于异步通信）**等。
40. 循环队列是**（将队列的存储空间看作一个环形结构）**，解决**（假溢出）**的问题，即队尾指针到达数组末尾后，可以回到数组头部继续存储元素。
41. 循环队列中，队空和队满的判断条件需要**（特殊处理）**，通常采用**（牺牲一个存储单元，即队尾指针始终指向一个空闲位置）**或**（设置标志位，记录队列中元素的个数）**的方法。

串

42. 串是**（由零个或多个字符组成的有限序列）**，也称**（字符串）**，是字符类型的数据结构，广泛应用于文本处理和信息检索等领域。
43. 串的存储结构有**（顺序存储）**和**（链式存储）**两种，类似于线性表，但由于串的元素是字符，所以通常采用顺序存储结构。
44. 串的基本操作包括**（求串长，计算串中字符的个数）**、**（串连接，将两个串连接成一个新串）**、**（串比较，比较两个串的大小关系）**、**（求子串，从串中提取指定位置和长度的子串）**和**（串替换，将串中的指定子串替换为另一个子串）**等。
45. 模式匹配是**（在主串中查找子串的过程）**，也称**（字符串匹配）**，是串的重要操作，广泛应用于文本搜索和病毒检测等领域。
46. BF算法是**（朴素的模式匹配算法）**，通过**（逐个比较主串和模式串的字符）**来实现，时间复杂度为O(m*n)，效率较低，但在模式串较短时仍然适用。
47. KMP算法是**（改进的模式匹配算法）**，利用**（模式串的next数组）**来减少不必要的回溯，时间复杂度为O(m+n)，效率较高，是模式匹配的首选算法。
48. 串的应用包括**（文本编辑，例如查找、替换和插入等操作）**、**（信息检索，例如搜索引擎和关键词匹配等）**和**（编译程序，例如词法分析和语法分析等）**。

数组和广义表

49. 数组是**（由类型相同的数据元素构成的有序集合）**，每个元素受n个线性关系的约束，是一种常用的数据结构，用于存储和处理大量相同类型的数据。

50. 数组的存储结构是**（顺序存储）**，通常采用**（行优先）或（列优先）**的方式存储，以方便访问任意位置的元素，存储空间是连续的。
51. 广义表是**（线性表的推广）**，允许表中元素本身又是一个广义表，是一种递归的数据结构，可以表示复杂的数据关系。
52. 广义表的元素可以是**（原子，即不可再分的元素）或（子表，即另一个广义表）**，体现了广义表的层次性和递归性。
53. 广义表的长度是指**（最外层包含的元素个数）**，例如广义表(a,(b,c),d)的长度为3。
54. 广义表的深度是指**（广义表展开后所含括号的最大层数）**，例如广义表(a,(b,c),d)的深度为2。

树

55. 树是**（n(n>=0)个结点的有限集合）**，当n=0时称为空树，是一种重要的非线性数据结构，用于表示具有层次关系的数据。
56. 树的基本术语包括**（根，树的顶端结点）、（叶子，没有子结点的结点）、（结点，树中的元素）、（深度，从根结点到该结点的路径长度）、（度，结点拥有的子结点个数）、（森林，多棵互不相交的树的集合）和（兄弟结点，具有相同父结点的结点）**等，是理解树结构的基础。
57. 树的存储结构有**（双亲表示法，记录每个结点的父结点）、（孩子表示法，记录每个结点的子结点）和（孩子兄弟表示法，记录每个结点的第一个子结点和下一个兄弟结点）**等，不同的存储结构适用于不同的应用场景。
58. 二叉树是**（每个结点最多有两个子树的树）**，子树有左右之分，是一种特殊的树，也是最常用的树结构之一。
59. 满二叉树是**（所有分支结点都存在左子树和右子树，并且所有叶子都在同一层的二叉树）**，是一种理想的二叉树，具有最高的效率。
60. 完全二叉树是**（除了最后一层外，每一层的结点数都达到最大值，并且最后一层的所有结点都集中在最左边的二叉树）**，可以利用数组进行存储，节省存储空间。
61. 二叉树的性质包括**（第i层最多有2^(i-1)个结点，深度为k的二叉树最多有2^k-1个结点，具有n个结点的完全二叉树的深度为log2n+1）**等，是分析二叉树算法的基础。
62. 二叉树的遍历方式有**（前序遍历，先访问根结点，然后访问左子树，最后访问右子树）、（中序遍历，先访问左子树，然后访问根结点，最后访问右子树）和（后序遍历，先访问左子树，然后访问右子树，最后访问根结点）**。
63. 前序遍历的顺序是**（根左右）**，常用于复制树结构。
64. 中序遍历的顺序是**（左根右）**，常用于二叉排序树的查找。
65. 后序遍历的顺序是**（左右根）**，常用于计算表达式树的结果。
66. 通过**（前序遍历和中序遍历）或（后序遍历和中序遍历）**可以唯一确定一棵二叉树，但只通过前序遍历和后序遍历无法唯一确定一棵二叉树。

67. 线索二叉树是**（在二叉树的结点中增加线索，指向前驱结点和后继结点）**，方便**（中序遍历）**，避免了递归遍历，提高了遍历效率，节省了空间。

68. 树和森林可以转换为**（二叉树）**进行处理，利用**（孩子兄弟表示法）**，将树转换为二叉树后，可以方便地使用二叉树的算法进行处理。

69. 哈夫曼树是**（带权路径长度最短的树）**，用于**（哈夫曼编码）**，可以有效压缩数据，常用于数据压缩和通信领域。

70. 树的应用包括**（文件系统，目录结构）**、**（数据库系统，索引结构）**和**（编译程序，语法树）**等，是计算机科学中重要的基础数据结构。

图

71. 图是**（由顶点的有穷非空集合和顶点之间边的集合组成）**，是一种比树更复杂的数据结构，用于表示多对多的关系。

72. 图分为**（有向图，边有方向）**和**（无向图，边没有方向）**，有向图的边称为弧，无向图的边称为边。

73. 图的基本术语包括**（顶点，图中的元素）**、**（边，顶点之间的连接）**、**（弧，有向图中的边）**、**（度，与顶点相连的边数）**、**（入度，指向顶点的边数）**、**（出度，从顶点指出的边数）**和**（路径，顶点之间的序列）**等，是理解图结构的基础。

74. 图的存储结构有**（邻接矩阵，用二维数组表示顶点之间的邻接关系）**和**（邻接表，为每个顶点建立一个链表，存储与该顶点相邻的顶点）**等，不同的存储结构适用于不同的图类型和应用场景。

75. 邻接矩阵是**（用一个二维数组来表示图中顶点之间的邻接关系）**，数组元素表示顶点之间是否存在边，以及边的权值，适用于稠密图，但空间复杂度较高。

76. 邻接表是**（为每个顶点建立一个单链表，存储与该顶点相邻的顶点）**，可以节省存储空间，适用于稀疏图，但查找邻接顶点需要遍历链表。

77. 图的遍历方式有**（深度优先搜索DFS，类似于树的前序遍历）**和**（广度优先搜索BFS，类似于树的层次遍历）**，是图算法的基础。

78. 深度优先搜索类似于**（树的前序遍历）**，从一个顶点开始，沿着一条路径尽可能深地搜索，直到到达最后一个顶点，然后回溯到上一个顶点，继续搜索其他路径。

79. 广度优先搜索类似于**（树的层次遍历）**，从一个顶点开始，依次访问与该顶点相邻的所有顶点，然后访问与这些顶点相邻的顶点，以此类推，直到访问完所有顶点。

80. 最小生成树是**（连接图中所有顶点，且总权值最小的树）**，是图论中的经典问题，常见的算法包括**（Prim算法和Kruskal算法）**。

81. Prim算法是**（从一个顶点开始，逐步加入与当前树相连的权值最小的边）**，每次选择与当前树距离最近的顶点，直到所有顶点都加入到树中，适用于稠密图。

82. Kruskal算法是**（每次选择权值最小的边，且不构成环）**，将所有边按权值从小到大排序，依次选择边，如果加入该边后不构成环，则将该边加入到树中，适用于稀疏图。
83. 最短路径是指**（图中两个顶点之间权值之和最小的路径）**，是图论中的另一个经典问题，常见的算法包括**（Dijkstra算法和Floyd算法）**。
84. Dijkstra算法是**（求解单源最短路径的算法）**，即从一个顶点到其他所有顶点的最短路径，时间复杂度为O(n^2)，适用于没有负权边的图。
85. Floyd算法是**（求解所有顶点对之间最短路径的算法）**，即任意两个顶点之间的最短路径，时间复杂度为O(n^3)，适用于有负权边的图。
86. 拓扑排序是**（对有向无环图进行排序，使得所有指向被排序顶点的顶点都排在前面）**，是一种线性排序，可以判断图中是否存在环。
87. 关键路径是**（AOE网中从源点到汇点具有最大路径长度的路径）**，影响工程完成的时间，是项目管理中的重要概念。
88. 图的应用包括**（社交网络，关系网络）**、**（交通网络，路线规划）**、**（计算机网络，路由算法）和（推荐系统，用户兴趣分析）**等，是计算机科学中重要的研究领域。

查找

89. 查找是在**（数据集合中寻找满足某种条件的数据元素的过程）**，是计算机科学中最基本的操作之一，广泛应用于各种应用场景。
90. 查找算法的评价指标是**（平均查找长度ASL，即查找成功时，查找次数的平均值）**，反映了查找算法的效率，ASL越小，查找效率越高。
91. 顺序查找是**（从线性表的一端开始，逐个比较关键字是否相等）**，时间复杂度为O(n)，适用于无序表，实现简单，但效率较低。
92. 二分查找是**（在有序表中，每次将待查区间缩小一半）**，时间复杂度为O(log2n)，效率较高，但要求数据必须有序，且存储结构必须是顺序存储。
93. 二分查找要求**（线性表必须是有序的，且采用顺序存储结构）**，是其局限性，不适用于链式存储结构和无序表。
94. 散列表是**（根据关键字直接访问内存存储位置的数据结构）**，通过**（散列函数）**计算存储地址，可以实现快速查找，时间复杂度接近O(1)。
95. 散列函数的设计原则是**（尽可能均匀地将关键字映射到存储空间）**，避免冲突，提高查找效率，常见的散列函数包括**（直接定址法、除留余数法和数字分析法）**等。
96. 解决散列冲突的方法包括**（开放定址法，寻找下一个空的散列地址）**、**（链地址法，将所有散列地址相同的关键字存储在同一个链表中）和（再散列法，使用多个散列函数）**等，不同的方法适用于不同的应用场景。

97. 开放定址法是指**（发生冲突时，寻找下一个空的散列地址）**，包括**（线性探测法、二次探测法和随机探测法）**等，容易产生堆积现象，影响查找效率。

98. 链地址法是指**（将所有散列地址相同的关键字存储在同一个链表中）**，可以有效解决冲突，但需要额外的存储空间存储链表指针。

99. 二叉排序树是**（左子树所有结点的值小于根结点的值，右子树所有结点的值大于根结点的值）**，是一种动态查找表，可以实现快速查找、插入和删除操作。

100. 平衡二叉树是**（左右子树的高度差不超过1的二叉排序树）**，可以有效提高查找效率，避免二叉排序树退化成链表，常见的平衡二叉树包括**（AVL树和红黑树）**等。

101. B树是一种**（多路平衡查找树）**，常用于**（文件系统和数据库系统）**中，可以有效减少磁盘I/O次数，提高查找效率。

排序

102. 排序是**（将一组数据按某种顺序排列的过程）**，是计算机科学中最基本的操作之一，广泛应用于各种应用场景。

103. 排序算法的评价指标包括**（时间复杂度，反映排序算法的效率）**、**（空间复杂度，反映排序算法的资源消耗）**和**（稳定性，反映排序算法对相等元素的处理方式）**等，是选择排序算法的重要依据。

104. 内部排序是指**（在排序过程中，所有数据都放在内存中）**，适用于数据量较小的情况，常见的内部排序算法包括**（插入排序、交换排序、选择排序、归并排序和基数排序）**等。

105. 外部排序是指**（在排序过程中，需要访问外存）**，适用于数据量非常大的情况，常见的外部排序算法包括**（归并排序）**，需要将数据分批读入内存进行排序，然后将排序结果合并。

108. 插入排序包括**（直接插入排序、折半插入排序和希尔排序）**等，都是基于插入思想的排序算法，适用于数据量较小或基本有序的情况。

109. 直接插入排序是**（将每个元素插入到已经排好序的序列中）**，时间复杂度为O(n^2)，实现简单，但效率较低，适用于数据量较小的情况。

110. 折半插入排序是**（利用二分查找来确定插入位置）**，减少了比较次数，但移动元素的次数不变，时间复杂度仍为O(n^2)，适用于数据量较小的情况。

111. 希尔排序是**（一种分组插入排序算法）**，通过**（设置不同的增量序列）**来将数据分组，然后对每组进行插入排序，时间复杂度与增量序列有关，通常优于直接插入排序。

112. 交换排序包括**（冒泡排序和快速排序）**等，都是基于交换思想的排序算法，通过不断交换元素的位置来实现排序。

113. 冒泡排序是**（通过不断交换相邻的逆序元素来逐步排序）**，时间复杂度为O(n^2)，实现简单，但效率较低，适用于数据量较小的情况。

114. 快速排序是**（选择一个基准元素，将序列分成两个子序列，递归排序）**，平均时间复杂度为O(nlog2n)，效率较高，是常用的排序算法之一，但最坏情况下时间复杂度为O(n^2)。

115. 选择排序包括**（简单选择排序和堆排序）**等，都是基于选择思想的排序算法，每次选择最小或最大的元素放在前面。

116. 简单选择排序是**（每次选择最小的元素放在前面）**，时间复杂度为O(n^2)，实现简单，但效率较低，适用于数据量较小的情况。

117. 堆排序是**（利用堆这种数据结构进行排序）**，时间复杂度为O(nlog2n)，效率较高，且空间复杂度较低，是一种常用的排序算法。

118. 归并排序是**（将两个或两个以上的有序序列合并成一个新的有序序列）**，时间复杂度为O(nlog2n)，效率较高，且稳定性较好，适用于数据量较大的情况。

119. 基数排序是**（根据关键字中各位的值来分配元素）**，不基于比较，时间复杂度为O(d(n+r))，适用于关键字位数较少的情况，例如对整数进行排序。

120. 各种排序算法的稳定性不同，稳定的排序算法可以保证**（相等元素的相对位置不变）**，例如归并排序和插入排序是稳定的，而快速排序和选择排序是不稳定的。

121. 排序算法的选择需要综合考虑**（数据规模、关键字分布和稳定性要求）**等因素，选择合适的排序算法可以提高程序的效率。

高级数据结构

122. 红黑树是一种**（自平衡二叉查找树）**，保证了在最坏情况下基本动态集合操作的时间复杂度为O(log n)，是一种常用的数据结构，广泛应用于各种应用场景。

123. B树是一种**（平衡的多路查找树）**，特别适用于磁盘等外部存储设备的数据组织，可以有效减少磁盘I/O次数，提高查找效率，常用于文件系统和数据库系统。

124. 堆**（优先队列）**是一种**（特殊的树形数据结构）**，每个节点都有一个优先级，常用于**（任务调度和事件处理）**，可以高效地找到优先级最高的元素。

125. 并查集是一种**（集合数据结构）**，用于（描述元素之间的分组关系），支持（查找元素所属集合和合并两个集合）的操作，常用于解决连通性问题。

算法设计策略

126. 递归是一种**（解决问题的方法）**，将问题分解为**（规模更小的相同问题的子问题）**，直到子问题可以直接求解，是一种常用的算法设计技巧。

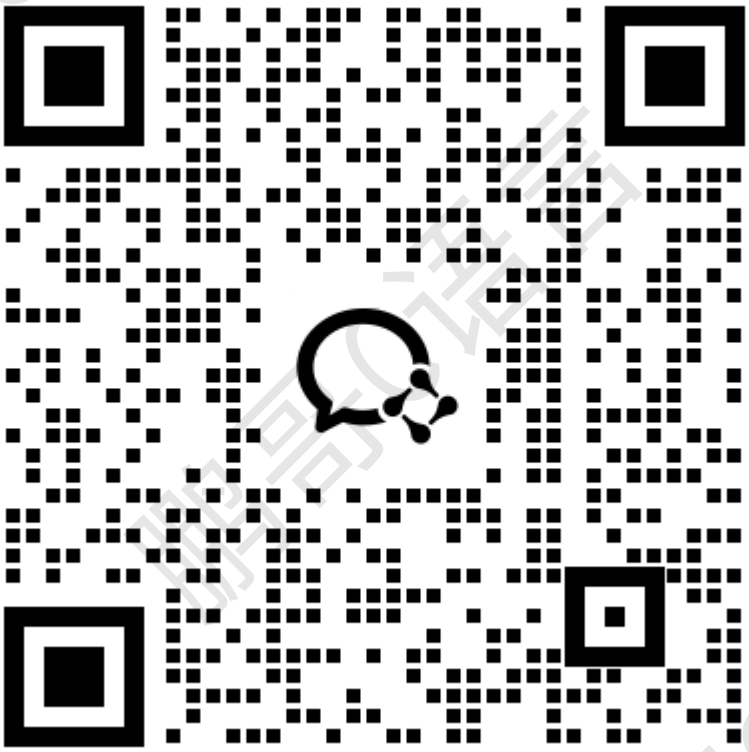
127. 分治法是**（将一个大问题分解成若干个规模较小的相同子问题）**，递归地解决这些子问题，然后将子问题的解合并得到原问题的解，适用于可以分解成独立子问题的情况，例如归并排序和快速排序。

128. 动态规划是**（将原问题分解为相互重叠的子问题）**，避免重复计算，存储子问题的解，从而高效地解决原问题，适用于具有最优子结构和重叠子问题性质的问题，例如背包问题和最长公共子序列问题。

129. 贪心算法是**（每一步都选择当前状态下最优的解）**，希望最终能够得到全局最优解，但不一定保证最优解，适用于具有贪心选择性质的问题，例如最小生成树算法和哈夫曼编码。

130. 回溯法是一种**（搜索问题的解的方法）**，通过**（试探和回溯）**来寻找所有可能的解，适用于（组合优化和排列生成）等问题，例如八皇后问题和旅行商问题。

131. 分支限界法是一种**（在解空间树上搜索问题解的方法）**，通过（剪枝）来减少搜索空间，提高搜索效率，适用于（组合优化）等问题。



三、操作系统原理 专升本高频考点背诵（极细化补充版）

（共145个高频考点+考前速记）

1. 操作系统是**（计算机系统中最核心的软件）**，它（**管理计算机的硬件和软件资源**），并（**提供用户与计算机硬件之间的接口，简化用户操作**）。

2. 操作系统的**（两大核心作用）**：**一是（管理计算机系统资源，如CPU、内存、I/O设备等，保证资源合理分配和高效利用），二是（为用户提供方便有效的服务，例如文件管理、进程管理等，简化用户与计算机的交互）**。
3. 操作系统的**（四个基本特征）**：**（并发性，指一段时间内多个程序同时运行）、（共享性，指系统资源可被多个并发执行的进程共同使用）、（虚拟性，指通过时分复用或空分复用等技术将一个物理实体变为多个逻辑实体）、（异步性，指进程以不可预知的速度向前推进，可能随时中断或暂停）**。
4. **（并发性）**是指**（宏观上，一段时间内多个程序似乎在同时运行），而（微观上，CPU在多个程序之间快速切换，交替执行，从而模拟出并行的效果）**。
5. **（共享性）**是指**（系统中的资源，例如CPU、内存、I/O设备等，可以被多个并发执行的进程共同使用，提高资源利用率）**。 共享方式分为（互斥共享）和（同时共享）两种。
6. **（虚拟性）**是指**（通过某种技术，例如时分复用或空分复用，将一个物理实体变为多个逻辑实体，让用户感觉拥有更多的资源）**。 常见的虚拟技术有（虚拟内存）和（虚拟设备）等。
7. **（异步性）**是指**（进程的执行不是完全顺序的，而是以不可预知的速度向前推进，可能随时中断或暂停，操作系统需要处理这种不确定性）**。
8. 操作系统的**（主要功能模块）**：（进程管理，负责进程的创建、撤销、调度和同步等）、（内存管理，负责内存的分配、回收、保护和虚拟内存的实现等）、（设备管理，负责设备的分配、驱动和控制等）、（文件管理，负责文件的存储、组织、检索和保护等）。
9. **（进程管理）**具体包括（进程控制，负责进程的创建、撤销和状态转换等）、（进程同步，负责协调多个进程的执行顺序，避免资源竞争）、（进程通信，负责进程之间的数据交换）、（处理机调度，负责选择下一个要执行的进程）。
10. **（内存管理）**具体包括（内存分配与回收，负责为进程分配和回收内存空间）、（地址映射，负责将逻辑地址转换为物理地址）、（内存保护与共享，负责保护内存空间不被非法访问，并允许多个进程共享内存）、（虚拟存储器，负责扩大内存容量，提高内存利用率）。
11. **（设备管理）**具体包括**（缓冲管理，负责管理I/O缓冲区，提高I/O效率）、（设备分配，负责为进程分配I/O设备）、（设备驱动，负责控制I/O设备的运行）**。
12. **（文件管理）**具体包括（文件存储空间管理，负责管理磁盘空间，分配和回收文件存储空间）、（目录管理，负责组织和管理文件目录）、（文件读写管理与保护，负责文件的读写操作和访问控制）。
13. **（系统调用）**是**（操作系统提供给应用程序使用的特殊接口，应用程序通过系统调用请求操作系统提供的服务，例如文件读写、进程创建等）**。 系统调用是（用户态程序进入核心态的唯一途径）。
14. **（用户态）和（核心态）**是（CPU的两种运行状态，用户态运行普通应用程序，核心态运行操作系统内核程序）。 核心态可以（执行特权指令），用户态（不能执行特权指令）。
15. **（中断）**是（CPU暂停当前任务，转而处理其他事件的机制，例如I/O完成、硬件故障等）。 中断分为（外部中断）和（内部中断）两种。

16. **（进程）**是（操作系统进行资源分配的最小单位，拥有独立的地址空间和系统资源）。 进程是（程序的一次执行过程）。

17. **（线程）**是**（CPU调度的最小单位，是进程中的一个执行流，共享进程的资源）**。 一个进程可以包含多个线程，（多线程可以提高程序的并发性）。

18. **（程序）**是**（指令的集合，描述了计算机要完成的任务）**，程序是（静态的，存储在磁盘上）。

19. 进程的**（三种基本状态）**：**（运行态，进程正在CPU上执行）**、**（就绪态，进程已经具备运行条件，等待CPU调度）**、**（阻塞态，进程因等待某个事件发生而暂停执行）**。

20. **（PCB，进程控制块）**是（进程存在的唯一标志，操作系统通过PCB来管理和控制进程）。 PCB记录了（进程的所有信息，例如进程ID、状态、优先级、资源使用情况等）。

21. **（进程同步）**是（控制多个进程按一定顺序执行，协调它们之间的资源访问，保证数据的一致性）。 进程同步的目的是（避免资源竞争和数据混乱）。

22. **（进程互斥）**是**（保证共享资源在同一时间只能被一个进程访问，避免多个进程同时修改共享资源导致数据错误）**。 进程互斥是（进程同步的一种特殊情况）。

23. **（临界资源）**是**（一次只能允许一个进程访问的资源，例如打印机、共享变量等）**。 访问临界资源需要（互斥地进行）。

24. **（临界区）**是**（访问临界资源的代码段，需要进行互斥访问控制）**。 临界区的访问需要（保证互斥性、有空让进、有限等待和让权等待）。

25. 解决互斥的**（软件方法，通过软件算法实现互斥，比较复杂且容易出错）**：包括（单标志法、双标志先检查法、双标志后检查法和Peterson算法）等。

26. 解决互斥的**（硬件方法，利用硬件指令实现互斥，效率高且可靠）**：包括（中断屏蔽方法，TestAndSet指令和Swap指令）等。

27. **（信号量）**机制用于**（进程同步与互斥，通过信号量的值来表示资源的可用数量）**。 信号量分为（整型信号量）和（记录型信号量）两种。

28. **（PV操作）**是**（信号量机制中的两种基本操作，P操作用于申请资源，V操作用于释放资源）**。 PV操作必须（成对出现，保证资源使用的正确性）。

29. P操作**（申请资源）**：(S = S - 1)，若(S < 0)，则表示资源不足，进程进入**（阻塞态，等待资源释放）**。

30. V操作**（释放资源）**：(S = S + 1)，若(S <= 0)，则表示有进程在等待资源，**（唤醒一个等待进程）**。

31. **（死锁）**是**（多个进程因竞争资源而互相等待的现象，导致所有进程都无法继续执行）**。 死锁是一种（资源竞争导致的僵局）。

32. 死锁产生的**（四个必要条件）**：**（互斥条件，资源必须以独占方式使用）**、**（请求与保持条件，进程在保持已分配资源的同时，又提出新的资源请求）**、**（不可剥夺条件，已分配给进程的资源不能被强制剥夺）**、**（循环等待条件，存在一个进程循环等待资源的链）**。

33. （死锁预防）通过（破坏死锁产生的四个必要条件之一，例如破坏请求与保持条件，可以避免死锁的发生，但可能会降低资源利用率）。

34. （死锁避免）在（**资源分配过程中，判断是否会导致死锁，如果会导致死锁，则拒绝分配资源，保证系统处于安全状态**）。 **常见的死锁避免算法是（银行家算法）**。

35. （死锁检测）在（系统运行过程中，定期检测系统中是否存在死锁，如果检测到死锁，则采取措施解除死锁）。

36. （处理机调度）分为（**高级调度、中级调度和低级调度，分别负责不同层次的资源分配**）。

37. （高级调度）又称（作业调度），决定（**哪些作业可以进入内存，准备运行，主要用于批处理系统**）。

38. （中级调度）又称（内存调度），决定（**哪些进程可以调入内存，哪些进程可以调出内存，提高内存利用率**）。 中级调度引入了（挂起状态）。

39. （低级调度）又称（进程调度），决定（哪个进程可以获得CPU，真正执行，是操作系统中最频繁的调度）。

40. （FCFS，先来先服务）调度算法（**按照作业或进程到达的先后顺序进行调度，简单易实现，但不利于短作业**）。

41. （SJF，短作业优先）调度算法（优先调度运行时间最短的作业或进程，可以有效缩短平均周转时间，但可能导致长作业饥饿）。

42. （优先级调度）算法（**按照进程的优先级进行调度，优先级高的先执行，可以灵活地满足不同进程的需求，但可能导致低优先级进程饥饿**）。 优先级可以是（静态的）或（动态的）。

43. （时间片轮转）调度算法（**将CPU时间划分成时间片，每个进程轮流执行一个时间片，适用于分时系统，可以提高响应速度**）。 时间片的大小（需要合理设置）。

44. （多级反馈队列调度算法）是（**时间片轮转调度算法的改进，设置多个就绪队列，每个队列的时间片大小不同，进程在不同队列之间移动，兼顾了长作业和短作业的需求**）。

45. （周转时间）是指（**作业从提交到完成所经历的时间，包括等待时间和运行时间**）。 **周转时间是（评价调度算法性能的重要指标）**。

46. （平均周转时间）是（所有作业周转时间的平均值，反映了系统的整体效率）。

47. （带权周转时间）是（周转时间与运行时间的比值，反映了作业的紧急程度）。

48. （平均带权周转时间）是（**所有作业带权周转时间的平均值，更公平地评价调度算法的性能**）。

49. （内存管理）的目的是（**提高内存利用率，为用户提供更大的可用内存空间，并方便用户使用内存**）。

50. （覆盖）技术用于（**解决早期计算机内存不足的问题，将不常用的程序段覆盖到内存中，节省内存空间**）。 **覆盖技术需要（程序员手动划分覆盖区）**。

51. （交换）技术用于（**提高内存利用率，将暂时不运行的进程调到外存，释放内存空间**）。 **交换技术需要（操作系统自动进行）**。

52. （连续分配）是指（为进程分配一块连续的内存空间，简单易实现，但容易产生碎片）。 连续分配包括（单一连续分配）和（分区分配）两种。

53. （内部碎片）是（分配给进程的内存空间中，未被利用的部分，例如固定分区分配中，进程大小小于分区大小）。

54. （外部碎片）是（内存中无法被利用的小块空闲空间，例如动态分区分配中，内存中存在很多小块空闲空间，但无法满足大进程的需求）。

55. （分页存储管理）将（内存空间划分成大小相等的页，进程也划分成大小相等的页，以页为单位进行分配和管理，可以有效利用内存空间，减少碎片）。

56. （页表）用于（实现逻辑地址到物理地址的映射，记录了每个页的物理地址）。 页表是（分页存储管理的关键数据结构）。

57. （快表TLB，Translation Lookaside Buffer）用于（加速地址转换，是一种高速缓存，存放了最近访问的页表项）。 快表可以（减少访问内存的次数，提高地址转换速度）。

58. （分段存储管理）将（内存空间划分成大小不等的段，进程也划分成大小不等的段，以段为单位进行分配和管理，可以方便程序的模块化管理，但容易产生外部碎片）。

59. （段表）用于（实现逻辑地址到物理地址的映射，记录了每个段的起始地址和段长）。 段表是（分段存储管理的关键数据结构）。

60. （虚拟内存）技术（允许程序加载部分代码和数据到内存中运行，其余部分保存在外存中，扩大了内存的容量，提高了内存利用率）。 虚拟内存是（现代操作系统的重要特征）。

61. （页面置换算法）用于（选择要换出的页面，当内存空间不足时，需要将某些页面换出到外存，为新页面腾出空间）。 页面置换算法的优劣（直接影响系统的性能）。

62. （OPT，最佳置换算法）是（理想的页面置换算法，选择未来最长时间内不会被访问的页面进行置换，可以达到最低的缺页率，但无法实现，因为无法预测未来）。

63. （FIFO，先进先出）页面置换算法（选择最先进入内存的页面进行置换，实现简单，但性能较差，容易产生Belady现象）。

64. （LRU，最近最少使用）页面置换算法（选择最近最少使用的页面进行置换，性能较好，接近OPT算法，但实现复杂，需要记录每个页面的访问时间）。

65. （CLOCK，时钟置换算法）是（LRU算法的一种近似实现，使用一个环形链表来表示内存中的页面，并使用一个指针指向当前页面，每次需要置换页面时，从当前指针开始扫描，根据页面的访问位和修改位来决定是否置换）。

66. （Belady现象）是指（在FIFO算法中，增加页面数反而导致缺页率增加的现象，说明FIFO算法不符合程序的局部性原理）。

67. （设备管理）的目标是（提高设备利用率，为用户提供方便的设备使用接口，并保证设备的安全性）。

68. （I/O控制方式）包括（程序直接控制、中断驱动方式、DMA方式和通道方式，这些方式的效率逐渐提高）。

69. （SPOOLING技术，Simultaneous Peripheral Operations On-Line）用于**（实现虚拟设备，将独占设备改造为共享设备，提高设备利用率）**。 例如（打印机共享）。

70. **（磁盘调度算法）**包括**（FCFS、SSTF、SCAN和CSCAN，用于优化磁盘的访问顺序，减少磁头的移动距离，提高磁盘I/O效率）**。

71. （FCFS，先来先服务）磁盘调度算法（按照磁盘I/O请求到达的先后顺序进行调度，简单易实现，但性能较差）。

72. **（SSTF，最短寻道时间优先）**磁盘调度算法**（优先调度距离当前磁头最近的磁盘I/O请求，可以减少平均寻道时间，但可能导致某些请求饥饿）**。

73. **（SCAN，扫描算法）**磁盘调度算法**（磁头在一个方向上移动，访问所有未完成的磁盘I/O请求，然后反向移动，避免了饥饿现象）**。 又称**（电梯算法）**。

74. （CSCAN，循环扫描算法）磁盘调度算法（磁头在一个方向上移动，访问所有未完成的磁盘I/O请求，然后快速返回到起始位置，减少了磁头移动的延迟）。

75. **（文件管理）**用于**（组织和管理计算机中的文件，提供文件的存储、检索、共享和保护等功能）**。

76. （文件系统）是（操作系统中用于管理文件的软件，负责文件的存储、组织、检索和保护等）。常见的文件系统有（FAT32、NTFS、EXT4等）。

77. **（文件）**是**（具有文件名的一组相关信息的集合，是计算机存储和管理数据的基本单位）**。 文件可以是（文本文件）或（二进制文件）。

78. （目录）是（用于组织和管理文件的结构，类似于书的目录）。 目录可以**（嵌套，形成树形目录结构）**。

79. **（绝对路径）**是从**（根目录开始的路径，唯一标识一个文件或目录）**。

80. （相对路径）是从（当前目录开始的路径，相对于当前目录的位置）。

81. **（文件类型）**包括**（文本文件，存储文本信息）**和**（二进制文件，存储二进制数据，例如图片、音频、视频等）**。

82. **（文件操作）**包括**（创建，创建新的文件）**、**（删除，删除已有的文件）**、**（读，从文件中读取数据）**、**（写，向文件中写入数据）**、**（修改，修改文件的内容）**等。**

83. （文件保护）通过**（访问控制列表ACL，Access Control List）实现，ACL记录了（哪些用户或组可以对文件进行哪些操作）**。

84. **（磁盘空闲空间管理）**的方法包括**（空闲表法、空闲链表法和位示图法，用于记录磁盘的空闲空间，方便文件系统的分配和回收）**。

85. **（索引文件）**通过（索引表）来提高文件访问速度，索引表中记录了（文件的物理地址）。 索引文件适用于（随机访问）。

86. （文件目录）用于（管理文件，记录文件的属性信息和存储位置）。

87. **（单级目录结构）**所有文件放在一个目录下，简单但不方便管理，容易出现**（文件名冲突）**。

88. （两级目录结构）分为（主目录）和（用户目录），每个用户拥有一个独立的目录，可以避免文件名冲突，但**（不利于文件共享）**。

89. **（树形目录结构）**是**（目前操作系统常用的目录结构，可以方便地组织和管理大量文件，支持多级目录和文件共享）**。

90. （文件共享）允许多个用户共享同一个文件，提高文件的利用率和协作效率。 文件共享需要**（考虑访问控制和数据一致性）**。

91. **（硬链接）**是指**（多个目录项指向同一个inode，inode是文件控制块，记录文件的元数据，例如大小、权限、创建时间等）**。 删除原始文件后，（硬链接仍然有效）。

92. **（软链接）**是指**（创建一个新的文件，其中包含指向原始文件的路径，类似于Windows中的快捷方式）**。 删除原始文件后，（软链接失效）。

93. **（作业）**是**（用户提交给计算机系统执行的任务，包括程序、数据和作业控制信息）**。

94. （脱机作业）是指（用户不能直接干预作业的执行，作业的执行过程由操作系统自动控制）。 适用于（批处理系统）。

95. **（联机作业）**是指**（用户可以与作业进行交互，例如通过键盘输入数据，查看输出结果）**。 适用于（分时系统）。

96. **（作业控制语言JCL）**用于（描述作业的执行步骤，例如编译、链接和运行等）。 JCL是（用户与批处理系统交互的接口）。

97. （批处理系统）适用于**（计算量大、交互性不强的作业，例如科学计算、数据处理等）**。 批处理系统（追求系统吞吐量）。

98. **（分时系统）**适用于**（需要频繁交互的作业，例如文本编辑、程序调试等）**。 分时系统（追求用户响应时间）。

99. （实时系统）适用于（需要快速响应的作业，例如工业控制、航空航天等）。 实时系统（追求响应的实时性和可靠性）。

100. **(PV操作)** 必须成对出现，有一个P操作就一定有一个V操作，保证**（资源使用的正确性）**。

101. (中断向量表) 存放**（中断服务程序的入口地址，当发生中断时，CPU根据中断向量表找到对应的中断服务程序）**。

102. **(DMA，Direct Memory Access)** 直接内存访问，允许**（外设直接和内存进行数据交换，不需要CPU参与，提高I/O效率）**。

103. **(通道)** 一种（特殊的处理器，可以独立地完成I/O操作，减轻CPU的负担）。

104. (虚拟设备) 通过（SPOOLING技术）将（独占设备改造为共享设备，提高设备利用率，例如打印机共享）。

105. (高速缓存Cache) 是（位于CPU和内存之间的小容量高速存储器，用于缓存常用的数据，提高CPU的访问速度）。 Cache基于（程序的局部性原理）。

106. **（进程调度算法的评价指标）**除了周转时间和带权周转时间外，还包括（CPU利用率，衡量CPU的繁忙程度）、（吞吐量，单位时间内完成的作业数量）和（公平性，保证每个进程都能获得合理的CPU时间）。

107. （实时调度算法）需要满足**（实时性的要求）**，分为**（硬实时调度算法，必须保证任务在截止时间内完成）**和**（软实时调度算法，允许偶尔错过截止时间）**。常见的实时调度算法有**（速率单调调度RMS）**和**（最早截止时间优先EDF）**。

108. **（多处理器调度）**需要考虑**（处理器之间的负载均衡，避免某些处理器过载，而另一些处理器空闲）**。常见的多处理器调度方法有**（静态分配）**和**（动态分配）**。

109. （影响缺页率的因素）包括**（页面大小，页面越大，缺页率越低，但内存利用率也越低）**、**（进程分配的物理页面数，页面数越多，缺页率越低）**和**（页面置换算法，不同的算法有不同的缺页率）**。

110. **（抖动，Thrashing）**是指**（进程频繁地进行页面置换，导致CPU大部分时间都用于页面置换，而不是执行程序，系统效率急剧下降）**。抖动的原因是**（进程分配的物理页面数太少，无法满足程序的需求）**。

111. **（工作集，Working Set）**是指**（进程在一段时间内频繁访问的页面的集合）**。操作系统可以**（根据工作集的大小来分配物理页面，避免抖动）**。

112. （文件系统的层次结构）从上到下依次为（用户接口层、文件目录层、存取控制层、逻辑文件系统层、物理文件系统层和设备驱动层）。

113. **（目录项）**记录了**（文件名和inode编号，用于快速查找文件）**。

114. （inode，索引节点）记录了**（文件的元数据，例如大小、权限、创建时间、修改时间、数据块指针等）**。inode是（文件系统的核心数据结构）。

115. **（文件分配方式）**包括**（连续分配、链接分配和索引分配，用于决定文件的数据块在磁盘上的存储方式）**。

116. （连续分配）将（文件的所有数据块存储在磁盘上的连续区域，优点是访问速度快，缺点是容易产生外部碎片，且不易扩展）。

117. **（链接分配）**将**（文件的每个数据块都包含指向下一个数据块的指针，优点是易于扩展，缺点是访问速度慢，需要顺序访问）**。

118. **（索引分配）**将（文件的所有数据块指针存储在一个索引块中，优点是支持随机访问，且易于扩展，缺点是需要额外的存储空间存储索引块）。

119. **（磁盘高速缓存，Disk Cache）**用于**（缓存磁盘上的数据，提高磁盘I/O速度）**。磁盘高速缓存可以**（减少磁盘的访问次数）**。

120. （RAID，Redundant Array of Independent Disks）是一种（磁盘阵列技术，通过将多个磁盘组合在一起，提高存储容量、性能和可靠性）。常见的RAID级别有（RAID 0、RAID 1、RAID 5等）。

121. **（磁盘阵列）**通过**（数据条带化、镜像和校验等技术）**来提高性能和可靠性。

122. （数据条带化）将（数据分割成多个小块，分别存储在不同的磁盘上，提高并行读写能力）。

123. **（镜像）**将**（数据复制到多个磁盘上，提高数据的可靠性，当一个磁盘损坏时，可以从其他磁盘恢复数据）**。

124. （校验）通过**（计算校验码来检测和纠正数据错误，提高数据的可靠性）**。

125. **（操作系统安全）**包括**（身份认证、访问控制、数据加密和审计跟踪等）**。

126. （身份认证）用于（验证用户的身份，防止非法用户访问系统）。 常见的身份认证方式有（密码认证、指纹识别和人脸识别等）。

127. **（访问控制）**用于**（限制用户对系统资源的访问权限，防止用户越权访问）**。 常见的访问控制模型有**（自主访问控制DAC、强制访问控制MAC和基于角色的访问控制RBAC）**。

128. （数据加密）用于（保护数据的机密性，防止数据被窃取或篡改）。 常见的数据加密算法有（对称加密算法和非对称加密算法）。

129. **（审计跟踪）**用于**（记录用户的操作行为，方便安全审计和故障排查）**。

130. **（病毒）**是一种**（恶意软件，可以自我复制和传播，破坏计算机系统）**。

131. （木马）是一种（伪装成正常程序的恶意软件，可以窃取用户的信息或控制用户的计算机）。

132. **（蠕虫）**是一种**（可以自我复制和传播的恶意软件，不需要用户干预）**。

133. （防火墙）是一种（网络安全设备，用于保护计算机系统免受网络攻击）。 防火墙可以（过滤网络流量，阻止恶意连接）。

134. **（入侵检测系统IDS）**用于**（检测计算机系统中的入侵行为，及时报警）**。

135. **（操作系统的发展趋势）**包括**（云计算、移动计算和物联网等）**。

136. （云计算）将（计算资源和服务通过网络提供给用户，用户可以按需使用，无需购买和维护硬件设备）。

137. **（移动计算**）是指**（在移动设备上进行的计算，例如智能手机和平板电脑）**。

138. **（物联网）**是指**（将各种物理设备连接到互联网，实现智能化管理和控制）**。

139. （微内核操作系统）将（操作系统的核心功能精简到最小，其他功能以用户态进程的方式运行，提高系统的可靠性和安全性）。

140. **（Linux）**是一种**（开源的操作系统内核，广泛应用于服务器、嵌入式设备和移动设备等）**。

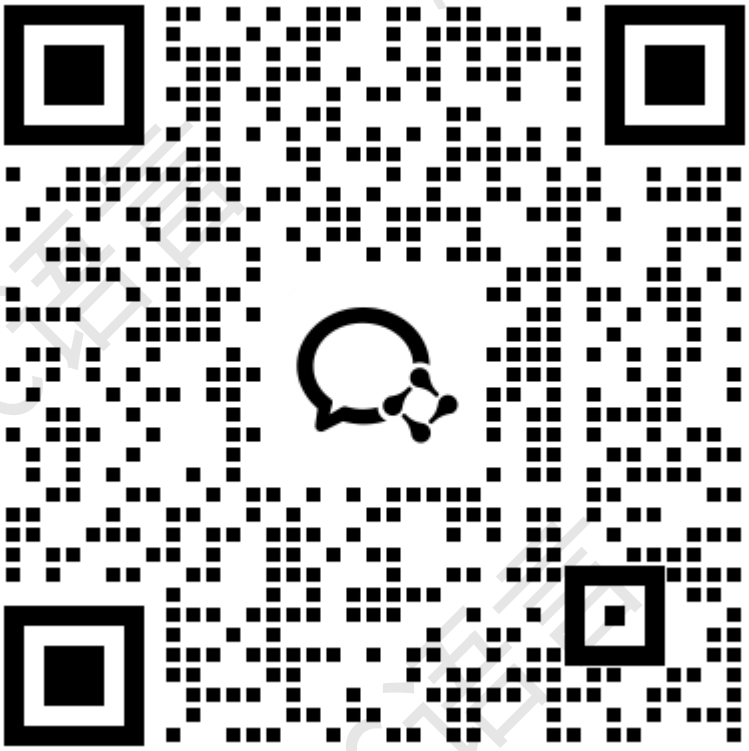
141. （Android）是一种**（基于Linux内核的移动操作系统，主要用于智能手机和平板电脑）**。

142. **(文件系统的一致性)** 需要通过**(日志)**或**(检查点)**等技术来保证，防止系统崩溃导致数据丢失或损坏。

143. **(虚拟化技术)** 允许在一台物理机上运行多个虚拟机，提高硬件利用率。 常见的虚拟化技术有**(VMware)**和**(KVM)**。

144. **(容器化技术)** 是一种轻量级的虚拟化技术，例如**(Docker)**，可以快速部署和运行应用程序。

145. (安全漏洞) 是**（操作系统或应用程序中的缺陷，可能被攻击者利用，导致安全问题）**。 需要及时**（打补丁）**来修复安全漏洞。



四、计算机网络 专升本高频考点背诵（极细化补充版）

（共112个高频考点+考前速记）

1. 计算机网络是**（指将地理位置分散的多台计算机，通过通信线路和通信设备互连起来，并按照一定的协议进行通信，以实现资源共享和信息传递的系统）**。 计算机网络的核心目标是**（使得网络中的计算机能够相互通信，共享硬件、软件和数据资源，提高计算机系统的整体效率和可靠性）**。
2. 计算机网络按照覆盖范围可以分为**（局域网（LAN, Local Area Network）、城域网（MAN, Metropolitan Area Network）和广域网（WAN, Wide Area Network））**。 不同的网络类型在**（覆盖范围、传输速率、延迟、成本和所使用的技术）**等方面都有显著差异。
3. **（局域网（LAN））**通常覆盖**（一个相对较小的地理区域，例如一个办公室、一栋建筑物、一个校园或一个家庭）**。 局域网的特点是**（传输速率高（通常在10Mbps到10Gbps之间），延迟低，成本相对较低，易于管理和维护，通常使用广播技术进行数据传输）**。

4. （城域网（MAN））覆盖（一个城市或城市的一部分）。 城域网通常由**（多个局域网互连而成，可以提供比局域网更广的覆盖范围和更高的带宽，但成本也更高，通常由电信运营商或大型企业建设和维护）**。

5. （广域网（WAN））覆盖**（一个非常大的地理区域，例如一个国家、一个地区或全球）**。 广域网通常使用**（长距离通信技术，例如光纤、微波、卫星等，传输速率相对较低，延迟较高，成本也最高，互联网是最大的广域网）**。

6. **计算机网络按拓扑结构**可以分为**（星型拓扑、环型拓扑、总线型拓扑、树型拓扑和网状型拓扑）**。 不同的拓扑结构在**（成本、可靠性、可扩展性、易于管理和维护）**等方面都有不同的优缺点，适用于不同的应用场景。

7. （星型拓扑）所有计算机都**（通过独立的线路连接到一个中心节点，中心节点通常是集线器或交换机）**。 星型拓扑的优点是**（易于安装、配置和管理，任何一条线路的故障都不会影响其他计算机的通信，易于扩展）**，缺点是**（中心节点是单点故障，中心节点故障会影响整个网络的通信，成本相对较高）**。

8. （环型拓扑）所有计算机都**（连接成一个闭合的环，数据沿着环的方向传输）**。 环型拓扑的优点是**（传输距离远，网络结构简单）**，缺点是**（任何一个节点或线路的故障都会影响整个网络的通信，维护困难，扩展性差，数据传输效率较低）**。

9. （总线型拓扑）所有计算机都**（通过一根公共的总线连接在一起，数据以广播的方式在总线上进行传输）**。 总线型拓扑的优点是**（结构简单，成本低）**，缺点是**（容易发生冲突，当多个计算机同时发送数据时，会发生冲突，导致数据传输失败，传输距离有限，网络性能随着计算机数量的增加而下降，维护困难）**。

10. （树型拓扑）是**（星型拓扑和总线型拓扑的结合，形成一种分层结构）**。 树型拓扑适用于**（具有分层结构的组织，例如大型企业或机构，易于扩展，易于管理和维护，但中心节点的故障会影响其下属的所有节点）**。

11. （网状型拓扑）所有计算机都**（相互连接，每个计算机都与多个其他计算机直接相连）**。 网状型拓扑的优点是**（可靠性高，即使某些线路或节点发生故障，数据仍然可以通过其他路径进行传输，安全性高）**，缺点是**（成本高，结构复杂，难以管理和维护，不适用于大型网络）**。

12. （OSI七层模型（Open Systems Interconnection Model））是（由国际标准化组织（ISO）提出的一个概念模型，将计算机网络的功能划分为七个不同的层次，每个层次负责不同的任务）。 OSI模型有助于（理解和设计网络协议，促进不同网络设备和协议之间的互操作性，简化网络问题的诊断和解决）。

13. OSI七层模型从上到下依次为**（应用层（Application Layer）、表示层（Presentation Layer）、会话层（Session Layer）、传输层（Transport Layer）、网络层（Network Layer）、数据链路层（Data Link Layer）和物理层（Physical Layer））**。

14. （物理层）负责**（在物理介质上透明地传输比特流（bit stream），即二进制数据）**。 物理层定义了（物理接口的特性（例如连接器的类型、引脚的数量和信号的电压）、传输介质的类型（例如双绞线、光纤和无线电波）、信号的编码方式（例如曼彻斯特编码和差分曼彻斯特编码）、数据传输速率等）。 物理层是**（OSI模型的最底层，是网络通信的基础）**。

15. （数据链路层）负责（在相邻的两个节点之间传输数据帧（frame），提供可靠的数据传输服务）。 数据链路层的主要功能包括（成帧（framing），将比特流组织成数据帧）、（物理地址寻址（physical addressing），使用MAC地址标识网络中的设备）、（差错控制（error control），检测和纠正数据传输中的错误）、（流量控制（flow control），防止发送方发送数据过快，导致接收方无法处理）、（介质访问控制（MAC, Media Access Control），控制多个设备对共享介质的访问）。 数据链路层使用（MAC地址（Media Access Control address），也称为物理地址或硬件地址，是网络设备的唯一标识符，由48位组成）。

16. （网络层）负责（在不同的网络之间传输数据包（packet），实现路由选择和拥塞控制）。 网络层的主要功能包括（逻辑地址寻址（logical addressing），使用IP地址标识网络中的设备）、（路由选择（routing），选择最佳的数据传输路径）、（拥塞控制（congestion control），避免网络拥塞，保证数据传输的可靠性）、（网际互连（internetworking），实现不同网络之间的互连互通）。 网络层使用（IP地址（Internet Protocol address），是网络设备的逻辑地址，由32位（IPv4）或128位（IPv6）组成）。

17. （传输层）负责（提供端到端的可靠或不可靠的数据传输服务，屏蔽底层网络的细节，为应用层提供统一的接口）。 传输层的主要协议包括（TCP（Transmission Control Protocol），提供面向连接的、可靠的数据传输服务）和（UDP（User Datagram Protocol），提供面向无连接的、不可靠的数据传输服务）。

18. （TCP协议）是（一种面向连接的、可靠的传输层协议，提供字节流服务）。 TCP协议通过（三次握手建立连接，确保双方都准备好进行数据传输）。 TCP协议通过（滑动窗口机制进行流量控制，防止发送方发送数据过快，导致接收方无法处理）。 TCP协议通过（拥塞控制机制避免网络拥塞，根据网络拥塞情况动态调整发送速率）。 TCP协议通过（序列号和确认号机制保证数据的可靠传输，确保数据按顺序到达，且没有丢失或损坏）。

19. （UDP协议）是（一种面向无连接的、不可靠的传输层协议，提供数据报服务）。 UDP协议的优点是（传输效率高，开销小，适用于实时性要求高的应用，例如音视频传输、在线游戏等）。 UDP协议的缺点是（不保证数据的可靠传输，可能会出现丢包、乱序和重复等问题）。

20. （会话层）负责（建立、管理和终止会话，提供会话管理服务，例如身份验证、授权和会话恢复）。 会话层通常与（应用层的具体实现相关，例如远程登录、文件传输和数据库访问等）。

21. （表示层）负责（数据格式转换、加密解密、压缩解压缩等，保证不同系统之间的数据可以正确交换）。 表示层的主要功能包括（数据编码（data encoding），将数据转换为网络传输的标准格式）、（数据加密（data encryption），保护数据的机密性）、（数据压缩（data compression），减少数据传输量）。

22. （应用层）负责（提供网络应用服务，直接与用户交互，是OSI模型的最顶层）。 应用层协议包括（HTTP（Hypertext Transfer Protocol），用于Web浏览器和Web服务器之间传输数据）、（FTP（File Transfer Protocol），用于在计算机之间传输文件）、（SMTP（Simple Mail Transfer Protocol），用于发送电子邮件）、（POP3（Post Office Protocol version 3），用于接收电子邮件）、（DNS（Domain Name System），用于将域名解析为IP地址）等。

23. （TCP/IP模型（Transmission Control Protocol/Internet Protocol Model））是（一个实际应用的模型，是互联网的基础，将计算机网络的功能划分为四个层次）。 TCP/IP模型更加简洁实用，被广泛应用于实际的网络通信中。
24. TCP/IP模型从上到下依次为（应用层（Application Layer）、传输层（Transport Layer）、网际层（Internet Layer）和网络接口层（Network Interface Layer））。
25. TCP/IP模型的（应用层）对应于OSI模型的（应用层、表示层和会话层，负责提供各种网络应用服务）。
26. TCP/IP模型的（网际层）对应于OSI模型的（网络层，负责实现路由选择和拥塞控制）。
27. TCP/IP模型的（网络接口层）对应于OSI模型的（数据链路层和物理层，负责在物理介质上透明地传输比特流）。
28. （IP地址（Internet Protocol address））是（互联网上设备的唯一标识，用于实现网络层寻址）。 IP地址类似于（电话号码，用于在网络中找到特定的设备）。
29. IP地址分为（IPv4（Internet Protocol version 4）和IPv6（Internet Protocol version 6））。
30. （IPv4地址）是（32位的地址，由四个字节组成，每个字节用十进制数表示，用点号分隔，例如192.168.1.100）。 IPv4地址已经（逐渐耗尽，无法满足日益增长的网络设备的需求）。
31. （IPv6地址）是（128位的地址，由八组十六进制数组成，每组用冒号分隔），例如（2001:0db8:85a3:0000:0000:8a2e:0370:7334）。 IPv6地址可以（提供更大的地址空间，解决IPv4地址短缺的问题）。
32. IPv4地址分为（A类地址、B类地址、C类地址、D类地址和E类地址）。 不同的地址类别适用于（不同规模的网络，用于进行网络规划和地址分配）。
33. （A类地址）的（第一个字节的范围是1-126，网络ID占一个字节，主机ID占三个字节）。 A类地址适用于（大型网络，可以支持大量的计算机）。
34. （B类地址）的（第一个字节的范围是128-191，网络ID占两个字节，主机ID占两个字节）。 B类地址适用于（中型网络，可以支持中等数量的计算机）。
35. （C类地址）的（第一个字节的范围是192-223，网络ID占三个字节，主机ID占一个字节）。 C类地址适用于（小型网络，可以支持少量的计算机）。
36. （D类地址）用于（多播（multicast），用于将数据发送给一组特定的计算机）。 D类地址的（第一个字节的范围是224-239）。
37. （E类地址）用于（实验（experimental），保留地址，不用于公共网络）。 E类地址的（第一个字节的范围是240-255）。
38. （私有IP地址）用于（内部网络，不能在互联网上直接使用，只能在局域网内部使用）。 私有IP地址可以（节省公有IP地址资源，提高网络安全性）。
39. 常用的私有IP地址段包括（10.0.0.0/8（即10.0.0.0到10.255.255.255）、172.16.0.0/12（即172.16.0.0到172.31.255.255）和192.168.0.0/16（即192.168.0.0到192.168.255.255））。

40. **（子网掩码（Subnet Mask））**用于**（划分IP地址的网络部分和主机部分，确定网络的大小）**。子网掩码与IP地址（进行与运算（AND operation），得到网络地址）。

41. **（网络地址（Network Address））**是**（IP地址与子网掩码进行与运算的结果，用于标识网络）**。网络地址用于（路由选择，路由器根据网络地址将数据包转发到目标网络）。

42. **（主机地址（Host Address））**是**（IP地址中除了网络地址以外的部分，用于标识网络中的设备）**。主机地址用于（在局域网内部找到特定的设备）。

43. **（默认网关（Default Gateway））**是**（连接内部网络和外部网络的路由器接口地址，用于将数据包发送到外部网络）**。默认网关类似于（邮局，用于将信件发送到其他城市）。

44. **（DNS（Domain Name System）域名系统）**用于**（将域名（例如www.example.com）解析为IP地址（例如192.0.2.1），方便用户记忆和访问）**。DNS服务器是（互联网的重要基础设施，提供域名解析服务）。

45. **（TCP（Transmission Control Protocol）协议）**是（一种面向连接的、可靠的传输层协议，提供字节流服务）。TCP协议通过（三次握手建立连接，四次挥手释放连接，滑动窗口机制进行流量控制，拥塞控制机制避免网络拥塞，序列号和确认号机制保证数据的可靠传输）。

46. **（TCP的三次握手）**用于**（建立可靠的TCP连接，确保双方都准备好进行数据传输）**。三次握手的过程是（客户端发送SYN（synchronize）报文，服务器收到后回复SYN+ACK（synchronize acknowledgement）报文，客户端收到后回复ACK（acknowledgement）报文，连接建立）。

47. **（TCP的四次挥手）**用于**（释放TCP连接，确保双方都完成了数据传输）**。四次挥手的过程是（客户端发送FIN（finish）报文，服务器收到后回复ACK报文，服务器发送FIN报文，客户端收到后回复ACK报文，连接释放）。

48. **（TCP的滑动窗口机制）**用于**（流量控制，防止发送方发送数据过快，导致接收方无法处理）**。滑动窗口的大小（由接收方的接收窗口和网络的拥塞窗口共同决定）。

49. **（TCP的拥塞控制机制）**用于**（避免网络拥塞，通过调整发送方的发送速率）**。拥塞控制算法包括（慢启动、拥塞避免、快速重传和快速恢复）。

50. **（UDP（User Datagram Protocol）协议）**是（一种面向无连接的、不可靠的传输层协议，提供数据报服务）。UDP协议的优点是（传输效率高，开销小，适用于实时性要求高的应用）。

51. **（HTTP（Hypertext Transfer Protocol）协议）**是**（一种应用层协议，用于在Web浏览器和Web服务器之间传输数据，基于TCP协议）**。HTTP协议是（Web应用的基础，用于请求和响应Web资源）。

52. **（HTTPS（Hypertext Transfer Protocol Secure）协议）**是（HTTP协议的安全版本，使用SSL/TLS进行加密，保护数据的机密性和完整性）。HTTPS协议可以（防止数据被窃听和篡改）。

53. **（FTP（File Transfer Protocol）协议）**是**（一种应用层协议，用于在计算机之间传输文件）**。FTP协议使用（TCP协议进行数据传输）。

54. **（SMTP（Simple Mail Transfer Protocol）协议）**是（一种应用层协议，用于发送电子邮件）。SMTP协议使用（TCP协议进行数据传输，端口号为25）。

55. **（POP3（Post Office Protocol version 3）协议）**是**（一种应用层协议，用于接收电子邮件）**。 POP3协议使用（TCP协议进行数据传输，端口号为110）。

56. **（DHCP（Dynamic Host Configuration Protocol）协议）**是（一种应用层协议，用于自动分配IP地址、子网掩码、默认网关和DNS服务器地址）。 DHCP协议可以（简化网络配置，提高网络管理效率）。

57. **（Telnet协议）**是**（一种应用层协议，用于远程登录计算机，以明文方式传输数据，安全性较低）**。

58. **（SSH（Secure Shell）协议）**是（一种安全的远程登录协议，使用加密技术保护数据的机密性和完整性）。 SSH协议可以（替代Telnet协议，提供安全的远程管理）。

59. **（SNMP（Simple Network Management Protocol）协议）**是**（一种应用层协议，用于网络管理，可以监控网络设备的运行状态，配置网络设备，进行故障诊断）**。

60. **（路由（Routing））**是指（选择数据包传输路径的过程，是网络层的主要功能）。 路由器是（实现路由的关键设备，根据路由表将数据包转发到目标网络）。

61. **路由算法**分为（静态路由和动态路由）。

62. **（静态路由）**由**（网络管理员手动配置，适用于小型网络，配置简单，但缺乏灵活性）**。

63. **（动态路由）**通过（路由协议自动学习和更新路由表，适用于大型网络，能够自动适应网络拓扑的变化）。

64. **常见的路由协议**包括**（RIP（Routing Information Protocol）、OSPF（Open Shortest Path First）**和BGP（Border Gateway Protocol））。

65. **（RIP（Routing Information Protocol）协议）**是一种（基于距离向量的路由协议，使用跳数作为度量值，每隔一段时间向邻居发送路由更新信息）。 RIP协议简单易实现，但（收敛速度慢，容易出现路由环路，不适用于大型网络）。

66. **（OSPF（Open Shortest Path First）协议）**是一种**（基于链路状态的路由协议，使用链路状态信息构建网络拓扑图，然后使用Dijkstra算法计算最短路径）**。 OSPF协议（收敛速度快，支持VLSM（Variable Length Subnet Masking），适用于大型网络）。

67. **（BGP（Border Gateway Protocol）协议）**是一种（用于自治系统（AS, Autonomous System）之间路由的协议，是互联网的核心路由协议）。 BGP协议（负责在不同的自治系统之间交换路由信息，保证互联网的互联互通）。

68. **（网络安全（Network Security））**是指（保护网络资源免受未经授权的访问、使用、修改、破坏或泄露，保证网络的可用性、完整性和机密性）。

69. **常见的网络安全威胁**包括**（病毒（virus）、木马（Trojan horse）、蠕虫（worm）、黑客攻击（hacking）、拒绝服务攻击（DoS, Denial of Service）等）**。

70. **（防火墙（Firewall））**是一种（网络安全设备，用于控制进出网络的流量，根据预先设定的规则允许或阻止特定的网络流量）。 防火墙可以（隔离内部网络和外部网络，阻止未经授权的访问，保护内部网络资源）。

71. （入侵检测系统（IDS, Intrusion Detection System））用于（检测网络中的恶意活动，例如黑客攻击、病毒传播等）。 IDS可以（实时监控网络流量，分析异常行为，及时发现和报告安全事件）。

72. （虚拟专用网络（VPN, Virtual Private Network））用于（在公共网络上建立安全的连接，将用户的网络流量加密传输，保护数据的机密性和完整性）。 VPN可以（用于远程访问内部网络资源，防止数据被窃听和篡改）。

73. （无线网络（Wireless Network））使用（无线电波进行通信，不需要物理线路连接，方便灵活）。 无线网络的优点是（灵活性高，易于部署），缺点是（安全性较低，容易受到攻击）。

74. 常见的无线网络标准包括（802.11a、802.11b、802.11g、802.11n、802.11ac和802.11ax，不同的标准在传输速率、频率和覆盖范围等方面都有所不同）。

75. （网络拓扑（Network Topology））是指（网络中计算机和其他设备的物理或逻辑排列方式，影响网络的性能、可靠性和可扩展性）。

76. （集线器（Hub））是（一种物理层设备，用于连接多台计算机，将接收到的数据广播到所有端口）。 集线器（只是简单地转发数据，不进行任何处理，容易发生冲突）。

77. （交换机（Switch））是（一种数据链路层设备，用于连接多台计算机，根据MAC地址转发数据，可以避免冲突）。 交换机可以（学习MAC地址，建立MAC地址表，提高数据传输效率）。

78. （路由器（Router））是（一种网络层设备，用于连接不同的网络，根据IP地址转发数据，实现路由选择）。 路由器可以（连接局域网和广域网，实现不同网络之间的通信）。

79. （网桥（Bridge））是（一种数据链路层设备，用于连接两个局域网，可以隔离冲突域）。 网桥可以（根据MAC地址转发数据，提高网络性能）。

80. （冲突域（Collision Domain））是指（网络中发生冲突的区域，当多个设备同时发送数据时，会发生冲突）。 冲突域越大，（发生冲突的可能性越高，网络性能越差）。

81. （广播域（Broadcast Domain））是指（网络中可以接收到广播消息的区域，广播消息会被发送到所有设备）。 广播域越大，（广播风暴的可能性越高，网络性能越差）。

82. （VLAN（Virtual LAN）虚拟局域网）用于（在逻辑上划分局域网，将一个物理局域网划分成多个虚拟局域网）。 VLAN可以（隔离广播域，提高网络安全性，简化网络管理）。

83. （网络管理（Network Management））包括（配置管理、故障管理、性能管理、安全管理和计费管理，用于保证网络的正常运行和高效利用）。

84. （网络协议分析器（Network Protocol Analyzer））用于（捕获和分析网络流量，可以查看数据包的内容，分析网络协议的运行情况，诊断网络故障）。 常用的网络协议分析器有（Wireshark）。

85. （Ping命令）用于（测试网络连通性，判断目标主机是否可达）。 Ping命令基于（ICMP协议）。

86. （Traceroute命令）用于（跟踪数据包的传输路径，显示数据包经过的所有路由器）。 Traceroute命令可以（帮助诊断网络故障，确定网络瓶颈）。

87. （网络拥塞（Network Congestion））是指（网络中的数据包数量超过了网络的处理能力，导致延迟增加和丢包，影响网络性能）。

88. （流量控制（Flow Control））用于（防止发送方发送数据过快，导致接收方无法处理，保证数据传输的可靠性）。

89. （拥塞控制（Congestion Control））用于（避免网络拥塞，通过调整发送方的发送速率，使网络达到最佳运行状态）。

90. （QoS（Quality of Service）服务质量）用于（保证不同应用的优先级，为高优先级应用提供更好的服务，例如语音和视频应用）。

91. （云计算（Cloud Computing））是一种（将计算资源和服务通过网络提供给用户的模式，用户可以按需使用，无需购买和维护硬件设备）。

92. （物联网（Internet of Things, IoT））是指（将各种物理设备（例如传感器、摄像头、家电等）连接到互联网，实现智能化管理和控制）。

93. （5G（5th Generation））是（第五代移动通信技术，具有高速率、低延迟和大连接的特点，可以支持更多的应用场景，例如自动驾驶、虚拟现实等）。

94. （软件定义网络（SDN, Software-Defined Networking））是一种（新型的网络架构，将控制平面和数据平面分离，实现网络的灵活控制和管理，可以简化网络配置，提高网络利用率）。

95. （网络功能虚拟化（NFV, Network Functions Virtualization））是一种（将网络功能（例如防火墙、路由器、负载均衡器等）虚拟化到通用硬件平台上的技术，降低网络设备的成本和复杂性，提高网络的灵活性和可扩展性）。

96. (无线局域网的安全协议)包括(WEP（Wired Equivalent Privacy）、WPA（Wi-Fi Protected Access）、WPA2（Wi-Fi Protected Access II）和WPA3（Wi-Fi Protected Access III）)。WPA2是目前最常用的无线安全协议，但WPA3提供了更高的安全性。

97. (网络地址转换（NAT, Network Address Translation）)用于(将私有IP地址转换为公有IP地址，解决IPv4地址短缺问题，允许多个设备共享一个公有IP地址访问互联网)。 NAT有多种类型，例如静态NAT、动态NAT和端口地址转换（PAT）。

98. (端口号（Port Number）)用于(标识计算机中的不同应用程序，区分不同的网络服务)。 端口号分为(知名端口（0-1023）、注册端口（1024-49151）和动态/私有端口（49152-65535）)。 常见的端口号有(HTTP:80, FTP:21, SMTP:25, DNS:53)。

99. (TCP的三次握手)用于(建立可靠的TCP连接，确保双方都准备好进行数据传输)。 三次握手的过程是(客户端发送SYN（synchronize）报文，服务器收到后回复SYN+ACK（synchronize acknowledgement）报文，客户端收到后回复ACK（acknowledgement）报文，连接建立)。

100. (TCP的四次挥手)用于(释放TCP连接，确保双方都完成了数据传输)。 四次挥手的过程是(客户端发送FIN（finish）报文，服务器收到后回复ACK报文，服务器发送FIN报文，客户端收到后回复ACK报文，连接释放)。

101. (数字签名（Digital Signature）)用于(验证数据的完整性和发送者的身份，防止数据被篡改和伪造)。 数字签名使用(非对称加密算法)。

102. **(数字证书（Digital Certificate）)** 用于(证明网站或个人的身份，由可信的第三方机构**（CA, Certificate Authority）颁发)**。 数字证书包含(公钥、身份信息和签名)。

103. **(对称加密（Symmetric Encryption）)** 使用(相同的密钥进行加密和解密，加密速度快，但密钥管理困难)。 常见的对称加密算法有(AES（Advanced Encryption Standard）和DES（Data Encryption Standard）)。

104. **(非对称加密（Asymmetric Encryption）)** 使用(不同的密钥进行加密和解密，公钥用于加密，私钥用于解密，密钥管理简单，但加密速度慢)。 常见的非对称加密算法有**(RSA)**。

105. **(HTTPS的加密过程)** 使用(对称加密和非对称加密相结合的方式，先使用非对称加密协商密钥**（例如使用Diffie-Hellman算法）**，然后使用对称加密进行数据传输，兼顾了安全性和效率)。

106. **(拒绝服务攻击（DoS, Denial of Service）)** 是一种(通过耗尽服务器资源，使其无法响应正常请求的攻击方式，导致服务器瘫痪)。 常见的DoS攻击有(SYN Flood攻击和UDP Flood攻击)。

107. **(分布式拒绝服务攻击（DDoS, Distributed Denial of Service）)** 是一种(利用多台计算机**（僵尸网络）**同时发起攻击的DoS攻击，攻击规模更大，更难防御)。

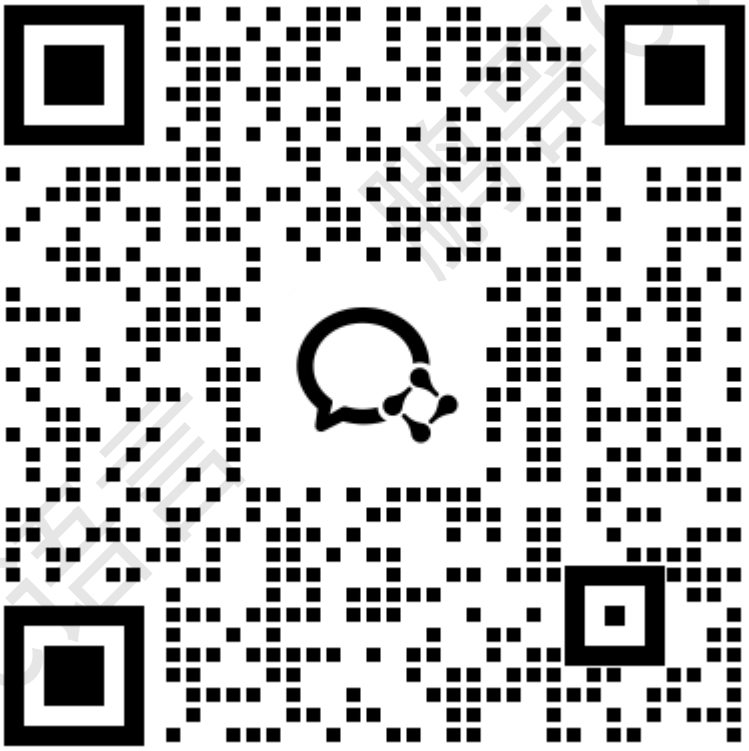
108. **(SQL注入攻击（SQL Injection Attack）)** 是一种**(通过在Web应用程序的SQL查询中插入恶意代码，来获取或修改数据库信息的攻击方式，例如绕过身份验证、窃取敏感数据)**。

109. **(跨站脚本攻击（XSS, Cross-Site Scripting）)** 是一种(通过在Web页面中插入恶意脚本，来窃取用户信息（例如Cookie）、篡改页面内容或重定向用户到恶意网站的攻击方式)。

110. **(ARP（Address Resolution Protocol）协议)** 用于**(将IP地址解析为MAC地址，在局域网内部实现寻址)**。

111. **(ICMP（Internet Control Message Protocol）协议)** 用于(在IP主机或路由器之间传递控制消息，例如错误报告、网络诊断)。 Ping命令就是基于ICMP协议实现的，用于测试网络连通性。

112. **(VPN的隧道协议)** 包括**(PPTP（Point-to-Point Tunneling Protocol）、L2TP（Layer 2 Tunneling Protocol）和IPSec（Internet Protocol Security）)**。 IPSec是目前最安全的VPN隧道协议，提供了数据加密、身份验证和完整性保护。



💡 需要更多C语言资料：如编译器、安装教程、C语言配套视频及刷题资料，扫描下方二维码添加助教领取

五、计算机组成原理 专升本高频考点背诵（极细化补充版）

（共105个高频考点+考前速记）

1. 计算机组成原理是**（一门研究计算机系统各个组成部分（如CPU、存储器、输入输出系统等）的功能、结构、以及它们之间如何相互连接和协同工作的学科）**。 计算机组成原理是**（计算机科学与技术专业的重要的专业基础课，是理解计算机系统底层工作原理，进行系统设计和优化的基础）**。
2. 计算机系统的五大部件包括**（运算器、控制器、存储器、输入设备和输出设备）**。 这五大部件是**（构成计算机系统的核心，任何计算机系统都必须包含这五个基本组成部分才能完成数据的处理和信息的交换）**。
3. **（运算器）**是**（计算机中执行算术运算（如加、减、乘、除）和逻辑运算（如与、或、非）的部件，是数据加工处理的中心）**。 运算器的核心是**（算术逻辑单元（ALU, Arithmetic Logic Unit），它能够执行各种算术和逻辑运算，是运算器的关键组成部分）**。
4. **（控制器）**是**（计算机的指挥中心，负责从存储器中取出指令，对指令进行译码，并发出控制信号，协调计算机各个部件的工作，控制指令的执行）**。 控制器由**（指令寄存器（IR, Instruction Register）、程序计数器（PC, Program Counter）、控制单元（CU, Control Unit）等组成，它们共同完成指令的控制和管理）**。

5. **（存储器）**是**（计算机中存储程序和数据的部件，是计算机系统记忆信息的场所）**。 存储器分为（主存储器（内存, Main Memory）和辅助存储器**（外存, Secondary Memory）**，主存储器用于存储当前正在运行的程序和数据，外存储器用于长期存储程序和数据）。

6. **（输入设备）**是**（将外部世界的信息转换为计算机可以识别和处理的数字形式的部件，是计算机获取信息的入口）**。 常见的输入设备有**（键盘、鼠标、扫描仪、摄像头、麦克风等，它们将用户的操作、图像、声音等信息转换为计算机可以处理的数字信号）**。

7. **（输出设备）**是**（将计算机处理后的数字信息转换为人类或其他设备可以理解的形式的部件，是计算机向外部世界展示结果的窗口）**。 常见的输出设备有**（显示器、打印机、音箱、投影仪等，它们将计算机处理后的数据以图像、文字、声音等形式呈现给用户）**。

8. **（冯·诺依曼体系结构（Von Neumann Architecture））**是**（现代计算机的基础，由数学家冯·诺依曼提出，定义了计算机的基本结构和工作方式）**。 冯·诺依曼体系结构的特点是（采用“存储程序”的工作方式，指令和数据以同等地位存储在存储器中，计算机按照指令的顺序从存储器中取出指令和数据进行处理，程序在执行过程中可以修改自身）。

9. （存储程序（Stored Program））是指**（将程序（指令序列）和数据都以二进制形式存储在存储器中，计算机可以自动地从存储器中取出指令和数据，通过运算器进行处理，并将结果存储回存储器，实现程序的自动化执行）**。 存储程序是（冯·诺依曼体系结构的核心思想，是现代计算机能够自动运行程序的基础）。

10. **（CPU（Central Processing Unit，中央处理器））**是**（计算机的核心部件，是计算机的运算和控制中心，负责指令的执行和数据的处理）**。 CPU包括（运算器和控制器，它们协同工作，完成指令的执行和数据的处理）。

11. （指令（Instruction））是（计算机执行的命令，是计算机完成特定操作的最小单位）。 指令由**（操作码（Opcode）**和地址码**（Address Code）**组成，操作码用于指定操作的类型（如加法、减法、数据传送等），地址码用于指定操作数（参与运算的数据）的存储地址或寄存器编号）。

12. **（指令系统（Instruction Set Architecture, ISA））**是**（计算机所能执行的所有指令的集合，是计算机硬件和软件之间的接口）**。 指令系统反映了（计算机的功能和性能，不同的计算机具有不同的指令系统，指令系统的设计直接影响计算机的效率和灵活性）。

13. 指令的寻址方式包括（立即寻址、直接寻址、间接寻址、寄存器寻址、寄存器间接寻址、基址寻址、变址寻址、相对寻址、堆栈寻址等，不同的寻址方式用于在存储器中定位操作数，影响指令的执行效率和灵活性）。

14. **（立即寻址（Immediate Addressing））**操作数**（直接包含在指令中，作为指令的一部分）**。 立即寻址的优点是（取数速度快，因为操作数就在指令中，不需要访问存储器），缺点是（操作数的大小受到指令长度的限制，只能表示较小的常数）。

15. **（直接寻址（Direct Addressing））**指令中**（直接给出操作数的存储器地址，通过该地址可以访问存储器中的操作数）**。 直接寻址的优点是（寻址简单，指令中直接包含操作数的地址），缺点是（寻址范围受到指令中地址码长度的限制，只能访问有限的存储器空间）。

16. **（间接寻址（Indirect Addressing））**指令中**（给出的是操作数地址的地址，即指令中包含的是一个指针，指向存储器中存储操作数地址的单元）**。 间接寻址的优点是**（可以扩大寻址范围，因为可以通过多次间接寻址访问更大的存储器空间）**，缺点是**（寻址速度慢，需要多次访问存储器才能找到操作数）**。

17. **（寄存器寻址（Register Addressing））**操作数（存放在CPU的寄存器中，指令中给出的是寄存器的编号）。 寄存器寻址的优点是（寻址速度快，因为寄存器位于CPU内部，访问速度非常快），缺点是（寄存器数量有限，只能存储少量的数据）。

18. **（存储器）**分为**（随机访问存储器（RAM, Random Access Memory）和只读存储器（ROM, Read-Only Memory），它们是计算机中两种主要的存储器类型，具有不同的特点和用途）**。

19. （RAM（随机访问存储器））可以**（随机地读取和写入数据，CPU可以快速访问RAM中的任何位置）**。 RAM是（易失性存储器，即断电后存储在RAM中的数据会丢失，因此RAM主要用于存储当前正在运行的程序和数据）。

20. **（ROM（只读存储器））**只能**（读取数据，不能随意写入数据，数据在生产时就被写入ROM中，之后只能读取）**。 ROM是（非易失性存储器，即断电后存储在ROM中的数据不会丢失，因此ROM主要用于存储计算机的启动程序（BIOS）和一些固定的数据）。

21. **（Cache（高速缓存））**是（位于CPU和主存储器之间的高速小容量存储器，用于缓存CPU经常访问的数据，以提高CPU访问存储器的速度）。 Cache的容量通常（远小于主存储器，但速度远高于主存储器）。

22. Cache的工作原理是**（利用程序的局部性原理（包括时间局部性和空间局部性），将CPU经常访问的数据存储在Cache中，当CPU需要访问数据时，首先在Cache中查找，如果找到（Cache命中），则直接从Cache中读取数据，如果找不到（Cache未命中），则从主存储器中读取数据，并将数据存储到Cache中，以便下次访问）**。

23. **Cache的命中率**是指**（CPU访问Cache时，找到所需数据的概率，是衡量Cache性能的重要指标）**。 命中率越高，（Cache的效率越高，CPU访问存储器的速度越快）。

24. Cache的替换算法包括（LRU（Least Recently Used，最近最少使用）、FIFO（First In First Out，先进先出）、LFU（Least Frequently Used，最不经常使用）等，用于决定当Cache已满时，替换哪个Cache块）。

25. **（主存储器）**的性能指标包括（存储容量、存取速度、存储周期等，这些指标直接影响计算机的整体性能）。

26. **（总线（Bus））**是**（计算机系统中各个部件之间传输信息的公共通道，是连接CPU、存储器和I/O设备的桥梁）**。 总线分为**（数据总线（Data Bus）、地址总线（Address Bus）和控制总线（Control Bus），它们分别用于传输数据、地址和控制信号）**。

27. （数据总线）用于（在CPU、存储器和I/O设备之间传输数据）。 数据总线的位数（也称为总线宽度，决定了计算机一次可以传输的数据量，例如32位数据总线一次可以传输4个字节的数据）。

28. **（地址总线）**用于（CPU向存储器或I/O设备发送地址信息，指定要访问的存储单元或I/O端口）。 地址总线的位数（决定了计算机可以访问的存储器空间，例如32位地址总线可以访问4GB的存储器空间）。

29. **（控制总线）**用于（传输控制信号，协调各个部件的工作，例如读写信号、中断请求信号等）。 控制总线（控制总线上的信号线数量和类型决定了CPU对各个部件的控制能力）。

30. 总线的仲裁方式包括**（集中仲裁和分布式仲裁，用于解决多个设备同时请求使用总线时，如何确定哪个设备获得总线使用权的问题）**。

31. **（集中仲裁）**由（一个中央仲裁器负责总线的分配，所有的设备都向中央仲裁器发出总线请求，由中央仲裁器根据一定的优先级策略决定哪个设备获得总线使用权）。 集中仲裁的优点是**（控制简单，易于实现）**，缺点是**（可靠性较低，如果中央仲裁器发生故障，整个总线系统将无法工作）**。

32. **（分布式仲裁）**由（各个部件竞争总线的使用权，每个设备都有自己的仲裁逻辑，通过相互协商来决定哪个设备获得总线使用权）。 分布式仲裁的优点是（可靠性高，即使某个设备发生故障，也不会影响其他设备对总线的使用），缺点是（控制复杂，实现成本较高）。

33. **（I/O接口（Input/Output Interface））**是**（CPU与外部设备之间的桥梁，用于实现CPU与外部设备之间的数据传输和控制）**。 I/O接口负责**（数据格式的转换（将CPU内部的数据格式转换为外部设备可以理解的格式，反之亦然）**、速度的匹配**（协调CPU和外部设备之间的速度差异，防止数据丢失或错误）**和控制信号的传递**（传递CPU发出的控制命令，以及外部设备的状态信息））**。

34. I/O的控制方式包括（程序查询方式、中断方式、DMA方式，不同的控制方式影响CPU的效率和I/O设备的响应速度）。

35. **（程序查询方式（Programmed I/O））**CPU（周期性地查询I/O设备的状态，判断I/O设备是否准备好进行数据传输，如果I/O设备准备好，则CPU进行数据传输，否则CPU继续查询）。 程序查询方式的缺点是（CPU需要花费大量的时间进行查询，效率低，CPU无法同时执行其他任务）。

36. **（中断方式（Interrupt-Driven I/O））**I/O设备（完成数据传输后，向CPU发出中断请求，CPU暂停当前程序的执行，转去执行中断服务程序，处理I/O设备的数据传输请求）。 中断方式可以（提高CPU的效率，CPU可以在I/O设备进行数据传输的同时执行其他任务，当I/O设备完成数据传输后，再通过中断请求通知CPU）。

37. **（DMA（Direct Memory Access，直接存储器访问）方式）**I/O设备（可以直接与存储器进行数据传输，不需要CPU的干预，CPU只需要在DMA传输开始前进行初始化设置，DMA传输完成后，CPU会收到DMA控制器的通知）。 DMA方式可以（进一步提高数据传输的效率，CPU可以完全освобожден от I/O数据传输的任务，专注于其他计算任务）。

38. **（中断（Interrupt））**是**（CPU暂停当前程序的执行，转去执行中断服务程序的过程，是实现I/O设备与CPU异步通信的重要机制）**。 中断用于（处理I/O设备发出的请求，例如数据传输完成、设备发生错误等）。

39. 中断的过程包括**（中断请求（I/O设备向CPU发出中断请求信号）**、中断响应**（CPU检测到中断请求信号后，判断是否允许中断，如果允许中断，则保存当前程序的上下文信息）**、中断处理**（CPU执行中断服务程序，处理I/O设备的请求）**和中断返回**（中断服务程序执行完成后，CPU恢复之前保存的程序上下文信息，继续执行被中断的程序）**）。

40. **（中断向量（Interrupt Vector））**是（中断服务程序的入口地址，用于快速定位中断服务程序，提高中断处理的效率）。 中断向量存储在（中断向量表中，每个中断向量对应一个特定的中断类型）。

41. **（指令流水线（Instruction Pipeline））**是**（将指令的执行过程分解为多个阶段（例如取指令、译码、执行、访存、写回），多个指令可以同时在不同的阶段执行，从而提高CPU的吞吐量）**。指令流水线可以（提高CPU的效率，使得CPU在单位时间内可以完成更多的指令）。

42. 指令流水线的阶段包括**（取指令（IF, Instruction Fetch）**：从存储器中取出指令、分析指令（ID, Instruction Decode）：对指令进行译码，确定指令的操作类型和操作数、执行指令**（EX, Execute）**：执行指令，进行算术或逻辑运算、访存**（MEM, Memory Access）**：如果指令需要访问存储器，则从存储器中读取数据或将数据写入存储器、写回**（WB, Write Back）**：将指令执行的结果写回寄存器）。

43. 指令流水线可能遇到的问题包括（结构冲突**（Structural Hazard）**、数据冲突**（Data Hazard）**和控制冲突**（Control Hazard）**，这些冲突会导致流水线暂停，降低CPU的效率）。

44. **（结构冲突（Structural Hazard））**是指**（多个指令同时访问同一个硬件资源，例如多个指令同时需要访问存储器）**。 解决结构冲突的方法包括（增加硬件资源，例如使用多个存储器端口，使得多个指令可以同时访问存储器）。

45. **（数据冲突（Data Hazard））**是指（指令的执行需要用到前面指令的结果，但前面指令的结果还没有准备好）。 解决数据冲突的方法包括（旁路技术（Bypassing）、流水线暂停（Pipeline Stall）、编译器优化（Compiler Optimization））。

46. **（控制冲突（Control Hazard））**是指（由于分支指令或中断等原因，导致流水线被打断，无法按照预定的顺序执行指令）。 解决控制冲突的方法包括**（分支预测（Branch Prediction）、延迟分支（Delayed Branch））**。

47. **（CISC（Complex Instruction Set Computer，复杂指令集计算机））**的特点是（指令数量多，指令格式复杂，寻址方式多，指令长度不固定，难以进行流水线优化）。 CISC的代表是（Intel的x86系列CPU，广泛应用于个人电脑和服务器）。

48. **（RISC（Reduced Instruction Set Computer，精简指令集计算机））**的特点是**（指令数量少，指令格式简单，寻址方式少，指令长度固定，易于进行流水线优化）**。 RISC的代表是（ARM系列CPU，广泛应用于移动设备和嵌入式系统）。

49. （多核处理器（Multi-Core Processor））是指（在一个芯片上集成多个CPU核心，每个核心都可以独立地执行指令）。 多核处理器可以**（提高计算机的并行处理能力，使得计算机可以同时执行多个任务，提高整体性能）**。

50. **（并行处理（Parallel Processing））**是指**（多个任务同时执行，可以提高计算机的效率）**。 并行处理可以（通过多核处理器、多处理器系统、集群系统等方式实现）。

51. **（存储器层次结构（Memory Hierarchy））**包括（Cache、主存储器和外存储器，它们按照速度和容量进行分层，形成一个金字塔结构）。 存储器层次结构用于（提高存储器的性能（速度）和降低存储器的成本（容量），使得计算机可以同时拥有高速和大容量的存储器）。

52. **（虚拟存储器（Virtual Memory））**是**（一种存储管理技术，允许程序使用比实际物理内存更大的地址空间，通过将程序的一部分存储在硬盘上，当需要时再加载到内存中）**。 虚拟存储器可以（提高内存的利用率，使得程序可以运行在内存不足的计算机上）。

53. **（地址转换（Address Translation））**是（将虚拟地址转换为物理地址的过程，使得程序可以使用虚拟地址访问存储器，而不需要关心物理地址的分配）。 地址转换由**（存储管理单元（MMU，Memory Management Unit）完成，MMU是CPU中的一个硬件部件，负责地址转换和存储器保护）**。

54. 常见的地址转换方式包括（分页存储管理和分段存储管理，它们是两种不同的虚拟存储器管理技术）。

55. **（分页存储管理（Paging））**将**（虚拟地址空间和物理地址空间都划分为大小相等的页（Page），程序按照页进行组织，存储器也按照页进行分配）**。 分页存储管理可以（提高内存的利用率，因为页的大小固定，可以减少存储碎片的产生，但容易产生内部碎片（Internal Fragmentation），即一个页中可能存在未被利用的空间）。

56. **（分段存储管理（Segmentation））**将**（虚拟地址空间划分为大小不等的段（Segment），程序按照逻辑模块进行划分，每个模块对应一个段）**。 分段存储管理可以（方便程序的模块化，易于实现共享和保护，但容易产生外部碎片（External Fragmentation），即存储器中存在一些小的空闲块，无法满足较大的程序段的存储需求）。

57. **（Cache的映射方式）**包括**（直接映射、全相联映射和组相联映射，不同的映射方式影响Cache的命中率和实现复杂度）**。

58. （直接映射（Direct Mapping））是指（每个主存块只能映射到Cache中的一个特定位置，Cache块的地址由主存块地址的一部分决定）。 直接映射的优点是**（实现简单，成本低）**，缺点是**（冲突率高，容易发生Cache抖动（Thrashing））**。

59. **（全相联映射（Fully Associative Mapping））**是指**（每个主存块可以映射到Cache中的任何位置，Cache块的地址不受主存块地址的限制）**。 全相联映射的优点是（冲突率低，可以充分利用Cache的空间），缺点是（实现复杂，成本高，需要使用复杂的查找算法）。

60. （组相联映射（Set Associative Mapping））是（直接映射和全相联映射的折中，将Cache分为若干组，每个主存块可以映射到Cache中的一个特定组中的任何位置）。 组相联映射可以（在冲突率和实现复杂度之间取得较好的平衡）。

61. （TLB（Translation Lookaside Buffer））是（**用于加速地址转换的Cache，存储了最近使用的虚拟地址和物理地址的映射关系**）。 TLB可以（**减少CPU访问页表的次数，提高地址转换的速度，从而提高程序的执行效率**）。 当CPU需要进行地址转换时，首先在TLB中查找，如果找到对应的映射关系（TLB命中），则直接使用该映射关系进行地址转换，如果找不到（TLB未命中），则需要访问页表，并将新的映射关系添加到TLB中。

62. （**总线宽度**）是指（总线一次可以传输的数据量，也称为总线位宽，是衡量总线性能的重要指标）。 总线宽度越大，（**数据传输速度越快，CPU与存储器或I/O设备之间的数据交换效率越高**）。 例如，32位总线一次可以传输4个字节的数据，而64位总线一次可以传输8个字节的数据。

63. （**总线时钟频率**）是指（总线的工作频率，也称为总线频率，是衡量总线性能的另一个重要指标）。 总线时钟频率越高，（**数据传输速度越快，总线在单位时间内可以传输更多的数据**）。 总线时钟频率通常以MHz或GHz为单位。

64. （**总线标准**）包括（**PCI（Peripheral Component Interconnect）、PCI-E（PCI Express）、USB（Universal Serial Bus）等，不同的总线标准具有不同的传输速度、接口类型和应用场景**）。 不同的总线标准（适用于不同的设备和应用，例如PCI-E主要用于连接显卡和高速存储设备，USB主要用于连接外部设备，如键盘、鼠标、打印机等）。

65. （硬盘（Hard Disk Drive, HDD））是（一种非易失性存储器，用于存储大量的数据，是计算机系统中主要的外部存储设备）。 硬盘的性能指标包括（**存储容量（硬盘可以存储的数据量）、转速（硬盘盘片的旋转速度，影响数据的读取速度）、平均寻道时间（磁头移动到目标磁道所需的时间，影响数据的读取速度）、缓存大小（硬盘内部的缓存，用于提高数据的读取速度）等**）。

66. （**固态硬盘（Solid State Drive, SSD）**）是（**一种基于闪存的存储设备，与传统的机械硬盘相比，具有读写速度快、抗震动、低功耗、无噪音等优点**）。 固态硬盘（逐渐取代传统的机械硬盘，成为计算机系统中的主流存储设备）。

67. （**RAID（Redundant Array of Independent Disks，独立磁盘冗余阵列）**）是一种（将多个硬盘组合起来，提高存储性能和可靠性的技术，通过将数据分布在多个硬盘上，可以提高数据的读取速度，通过数据冗余备份，可以提高数据的可靠性）。 RAID有多种级别（例如RAID 0、RAID 1、RAID 5、RAID 10等），不同的级别具有不同的性能和可靠性特点。

68. （存储器的校验码）用于（检测和纠正存储器中的错误，保证数据的完整性和可靠性）。 常见的校验码包括（奇偶校验码（Parity Check Code）、海明码（Hamming Code）和CRC校验码（Cyclic Redundancy Check））。

69. （**奇偶校验码**）可以（**检测一位错误，但不能纠正错误，只能判断数据是否出错，无法确定出错的位置**）。 奇偶校验码（**通过在数据中添加一个校验位，使得数据中1的个数为奇数或偶数，从而检测数据是否出错**）。

70. （海明码）可以（检测和纠正一位错误，同时还可以检测两位错误，但不能纠正两位错误）。 海明码（通过添加多个校验位，使得每个校验位可以检测一组特定的数据位，从而确定出错的位置，并进行纠正）。

71. **（CRC校验码）**可以**（检测多位错误，是一种常用的数据校验码，广泛应用于数据通信和存储领域）**。CRC校验码**（通过将数据看作一个多项式，然后用一个特定的生成多项式进行除法运算，得到的余数作为校验码，用于检测数据是否出错）**。

72. **（浮点数（Floating-Point Number））**用于**（表示实数，包括整数和小数，是计算机中常用的数据类型）**。浮点数的表示方式包括**（IEEE 754标准，该标准定义了浮点数的格式、精度和运算规则）**。

73. （指令周期（Instruction Cycle））是指（CPU执行一条指令所需要的时间，是衡量CPU性能的重要指标）。指令周期包括（取指令周期（Fetch Cycle）、分析指令周期（Decode Cycle）和执行指令周期（Execute Cycle），不同的指令可能需要不同的执行周期）。

74. **（时钟周期（Clock Cycle））**是**（CPU的基本时间单位，也称为节拍周期，是CPU内部各种操作的最小时间间隔）**。时钟周期越短，**（CPU的运行速度越快，单位时间内可以完成更多的操作）**。

75. （机器周期（Machine Cycle））是（CPU完成一个基本操作所需要的时间，例如读取存储器、写入存储器等）。机器周期通常由（多个时钟周期组成，一个机器周期通常包含多个时钟周期）。

76. **（汇编语言（Assembly Language））**是**（一种面向机器的程序设计语言，使用助记符表示指令，而不是二进制代码）**。汇编语言可以**（直接控制硬件，编写效率高，但可读性差，依赖于特定的硬件平台）**。

77. （微程序控制器（Microprogrammed Controller））使用（微程序控制指令的执行，将每条指令分解为一系列微操作，每个微操作对应一条微指令，微指令存储在控制存储器中）。微程序控制器的优点是（设计灵活，易于修改和扩展），缺点是（速度较慢，因为需要从控制存储器中读取微指令）。

78. **（硬布线控制器（Hardwired Controller））**使用（硬件逻辑电路控制指令的执行，直接使用逻辑门电路实现指令的控制逻辑）。硬布线控制器的优点是**（速度快，因为不需要从存储器中读取指令）**，缺点是**（设计复杂，难以修改和扩展）**。

79. **（流水线深度（Pipeline Depth））**是指**（流水线中阶段的数量，流水线深度越大，可以将指令的执行过程划分得更细，从而提高CPU的吞吐量）**。流水线深度越大，**（理论上的吞吐量越高，但实际效果会受到各种因素的影响，例如流水线冲突、分支预测错误等）**。

80. （超标量技术（Superscalar Technology））是指**（在一个时钟周期内可以执行多条指令，通过增加CPU内部的执行单元，使得CPU可以并行执行多个任务）**。超标量技术可以（提高CPU的并行处理能力，从而提高CPU的性能）。

81. **(Cache的写策略)**包括(写直通**（Write-Through）**和写回**（Write-Back）**，用于决定当CPU向Cache中写入数据时，如何更新主存储器中的数据)。

82. (写直通（Write-Through）) 指(数据同时写入Cache和主存，保证Cache和主存中的数据一致)。写直通的优点是（实现简单，数据一致性好），缺点是（每次写操作都需要访问主存，速度较慢）。

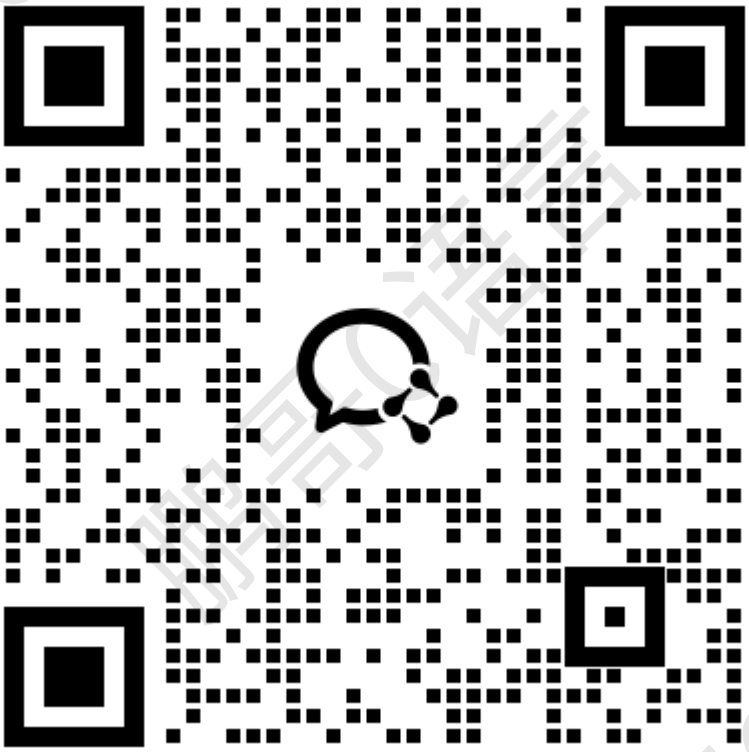
83. **(写回（Write-Back）)** 指**(数据只写入Cache，当Cache块被替换时才写回主存，Cache块中有一个修改位（Dirty Bit），用于标记该Cache块是否被修改过)。** 写回的优点是（写操作速度快，因为不需要每次都访问主存），缺点是（实现复杂，数据一致性风险较高，需要使用Cache一致性协议）。
84. (Cache一致性问题) 是指(多核处理器中，多个Cache副本的数据不一致，需要使用Cache一致性协议来保证数据的一致性)。
85. **(MESI协议)** 是一种**(常用的Cache一致性协议，用于保证多核处理器中Cache的数据一致性，MESI代表四种Cache状态：Modified、Exclusive、Shared、Invalid)。**
86. (总线仲裁方式) 包括(集中式仲裁和分布式仲裁，用于解决多个设备同时请求使用总线时，如何确定哪个设备获得总线使用权的问题)。
87. **(集中式仲裁)** 包括**(链式查询、计数器定时查询和优先级判优，不同的仲裁方式具有不同的优先级策略和实现复杂度)。**
88. **(DMA传输方式)** 包括(停止CPU访问主存、周期挪用和DMA与CPU交替访问主存，不同的传输方式对CPU的影响不同)。
89. **(中断优先级)** 用于**(确定CPU响应中断的顺序，当多个设备同时发出中断请求时，CPU会优先响应优先级较高的中断)。**
90. **(向量中断)** 通过(中断向量表查找中断服务程序的入口地址，提高中断处理的效率)。
91. **(总线带宽)** 指**(总线在单位时间内可以传输的数据量，是衡量总线性能的重要指标)。**
92. (总线标准) 包括(ISA、EISA、PCI、AGP和PCI-E，不同的总线标准具有不同的传输速度和接口类型)。
93. **(存储器的编址方式)** 包括**(字编址和字节编址，不同的编址方式影响存储器的访问效率)。**
94. (虚拟存储器的页面置换算法) 包括**(FIFO、LRU、OPT和Clock，不同的置换算法影响虚拟存储器的性能)。**
95. **(指令格式)** 包括(操作码字段和地址码字段，用于描述指令的结构和功能)。
96. (寻址方式) 用于(确定操作数在存储器中的地址，不同的寻址方式影响指令的执行效率和灵活性)。
97. **(CPU的性能指标)** 包括**(主频、CPI、MIPS和FLOPS，用于衡量CPU的性能)。**
98. **(主频)** 指(CPU的时钟频率，是CPU的基本时间单位)。
99. **(CPI（Cycles Per Instruction）)** 指**(执行一条指令所需要的时钟周期数，CPI越小，CPU的效率越高)。**
100. (MIPS（Million Instructions Per Second）) 指(每秒执行百万条指令数，是衡量CPU性能的指标之一)。
101. **(FLOPS（Floating-point Operations Per Second）)** 指(每秒执行浮点运算数，是衡量CPU浮点运算能力的指标)。

102. **(多处理器系统)** 包括(对称多处理器（**SMP, Symmetric Multiprocessing**）和非对称多处理器（**AMP, Asymmetric Multiprocessing**），不同的多处理器系统具有不同的体系结构和应用场景)。

103. **(集群系统（Cluster System）)** 是(一组相互连接的计算机，作为一个统一的资源来使用，用于提高计算能力和可靠性)。

104. **(网格计算（Grid Computing）)** 是一种**(分布式计算模式，利用互联网上的闲置计算资源，用于解决大规模的计算问题)**。

105. **(云计算（Cloud Computing）)** 是一种**(将计算资源和服务通过网络提供给用户的模式，用户可以按需使用，无需购买和维护硬件设备)**。



六、数据库系统 专升本高频考点背诵（极细化补充版）

（共105个高频考点+考前速记）

1. 数据库是**（长期存储在计算机内的、有组织的、可共享的数据集合，它不仅仅是数据的简单堆砌，而是经过精心设计和组织，能够反映现实世界实体及其关系的结构化数据集合）**。 数据库的目的是**（高效地存储和管理数据，支持各种应用，例如企业管理系统、电商平台、社交网络等，这些应用都需要依赖数据库来存储和管理大量的数据）**。
2. 数据库管理系统（DBMS）是（**管理和控制数据库的软件系统，它是用户与数据库之间的接口，负责数据的存储、检索、更新和安全管理）**。 DBMS提供（数据定义（DDL，用于定义数据库的结构，例如创建表、定义字段等）、数据操作（DML，用于对数据库中的数据进行增删改查操作）、数据控制（DCL，用于控制用户对数据库的访问权限和事务管理）和数据维护等功能（例如备份、恢复、性能优化等））。
3. 数据库系统（DBS）由**（数据库、数据库管理系统、应用系统、数据库管理员（DBA）和用户组成，这五个部分共同构成了一个完整的数据库系统）**。 数据库系统是（一个复杂的系统，需要各个组成部分协同工作，才能保证数据的正确性、一致性和安全性，并为用户提供高效的数据服务）。
4. 数据模型是**（对现实世界数据特征的抽象和描述，它定义了数据的组织方式、数据之间的关系以及数据的约束条件）**。 常用的数据模型有**（层次模型（以树状结构表示数据及其关系）、网状模型（以网状结构表示数据及其关系）、关系模型（以二维表的形式表示数据及其关系，是目前最流行的数据模型）**和**面向对象模型（以对象的形式表示数据及其关系））**。
5. 关系模型是（目前最流行的数据模型，它使用关系（表）来表示数据和数据之间的联系，每个关系对应一个二维表，表中的每一行表示一个元组（记录），每一列表示一个属性（字段））。 关系模型的优点是（简单、易于理解和使用，它使用简单直观的二维表来表示数据，易于被用户理解和接受，并且关系模型具有良好的数学基础，可以进行严格的理论分析和优化）。
6. 关系由（关系模式和关系实例组成，关系模式是对关系的静态描述，关系实例是关系在某一时刻的动态数据）。 关系模式是（对关系的描述，包括关系名、属性名和属性的域，它定义了关系的结构和约束条件），关系实例是（关系在某一时刻的具体数据，它是关系模式的具体实现，随着数据的增删改查，关系实例会不断变化）。
7. 关系模式包括（关系名**（用于标识关系）**、属性名**（用于描述关系的特征）**和属性的域（**属性的取值范围，例如整数、字符串、日期等）**，这些元素共同定义了关系的结构）。 属性的域是（属性的取值范围，它限制了属性可以取的值，保证数据的有效性和一致性，例如年龄属性的域可以是0到150之间的整数）。
8. 关系具有（五个基本性质：**属性的原子性（属性的值不可再分）**、**关系的无序性（元组的顺序无关紧要）**、**属性的唯一性（关系中不能有相同的属性名）**、**元组的无序性（元组的顺序无关紧要）**和**元组的唯一性（关系中不能有完全相同的元组）**）。 这些性质（保证了关系的规范性和数据的完整性）。
9. 关系数据库是（基于关系模型建立的数据库，它使用关系来组织和存储数据，并使用SQL语言进行数据操作）。 关系数据库使用（SQL语言进行数据操作，SQL是关系数据库的标准查询语言，可以进行数据查询、数据更新、数据定义和数据控制等操作）。

10. SQL（Structured Query Language）是**（结构化查询语言，是关系数据库的标准查询语言，用于访问和管理关系数据库中的数据）**。SQL可以**（进行数据查询（SELECT语句）、数据更新（INSERT、UPDATE和DELETE语句）、数据定义（CREATE、ALTER和DROP语句）**和**数据控制等操作（GRANT和REVOKE语句）**，它可以满足用户对数据库的各种需求）。
11. SQL的组成部分包括（**数据查询语言（DQL，用于查询数据）**、**数据操作语言（DML，用于操作数据）**、**数据定义语言（DDL，用于定义数据库结构）**和**数据控制语言（DCL，用于控制数据库访问权限和事务管理）**，这四个部分共同构成了完整的SQL语言）。
12. 数据查询语言（DQL）用于**（查询数据库中的数据，主要使用SELECT语句，SELECT语句可以从一个或多个表中检索数据，并可以进行过滤、排序、分组等操作）。**
13. 数据操作语言（DML）用于（对数据库中的数据进行操作，包括INSERT（插入数据）、UPDATE（更新数据）和DELETE（删除数据）语句，这些语句可以修改数据库中的数据）。
14. 数据定义语言（DDL）用于（定义数据库的结构，包括**CREATE（创建数据库对象，例如表、索引、视图等）**、**ALTER（修改数据库对象）**和**DROP（删除数据库对象）**语句，这些语句可以改变数据库的结构）。
15. 数据控制语言（DCL）用于（控制数据库的访问权限和事务，包括GRANT（授予用户访问权限）和REVOKE（撤销用户访问权限）语句，这些语句可以保证数据库的安全性）。
16. 关系代数是**（一种抽象的查询语言，用于描述对关系的操作，它使用一系列运算符来操作关系，得到新的关系）**。关系代数的操作包括（并（UNION）、差（DIFFERENCE）、交（INTERSECTION）、笛卡尔积（CARTESIAN PRODUCT）、选择（SELECT）、投影（PROJECT）、连接（JOIN）和除（DIVISION），这些操作是关系代数的基本操作）。
17. 选择操作是**（从关系中选择满足条件的元组，使用σ符号表示，例如σage>20(Student)表示从Student关系中选择年龄大于20的元组）。**
18. 投影操作是**（从关系中选择指定的属性，使用π符号表示，例如πname,age(Student)表示从Student关系中选择name和age属性）。**
19. 连接操作是（将两个关系中满足连接条件的元组组合在一起，常用的连接包括等值连接（EQUIJOIN）、自然连接（NATURAL JOIN）和外连接（OUTER JOIN），连接操作可以实现关系的组合）。
20. 数据库设计是将（现实世界的需求转换为数据库的结构的过程，它是一个迭代的过程，需要不断地调整和优化）。数据库设计包括**（需求分析、概念设计、逻辑设计和物理设计，这四个阶段是数据库设计的核心步骤）。**
21. 需求分析是（了解用户的需求，确定数据库需要存储哪些数据和支持哪些功能，它是数据库设计的第一步，也是最重要的一步，需求分析的质量直接影响数据库设计的质量）。
22. 概念设计是**（建立一个独立于具体DBMS的概念模型**，常用的概念模型是实体-联系模型（E-R模型），概念模型是数据库设计的基础，它描述了现实世界实体及其关系）。
23. 逻辑设计是（将概念模型转换为具体的数据库模型，例如关系模型，逻辑模型是数据库设计的关键步骤，它决定了数据库的结构和组织方式）。

24. 物理设计是**（确定数据库的物理存储结构，例如索引、存储方式等，物理模型是数据库设计的最后一步，它决定了数据库的性能和效率）。**

25. 实体-联系模型（E-R模型）是（一种用于描述现实世界概念模型的图形化工具，它使用实体、属性和联系来表示现实世界中的对象及其关系）。E-R模型包括（实体、属性和联系，这三个要素是E-R模型的基本组成部分）。

26. 实体是**（现实世界中可以区分的对象，例如学生、课程、教师等，每个实体都具有唯一的标识符）。**

27. 属性是**（实体的特征，例如学生的姓名、学号、年龄等，每个属性都有一个取值范围）。**

28. 联系是**（实体之间的关系，例如学生选修课程、教师教授课程等，联系可以是一对一、一对多或多对多）。**

29. 联系的类型包括（一对一联系（一个实体只能与另一个实体相关联）、一对多联系（一个实体可以与多个实体相关联，而另一个实体只能与一个实体相关联）和多对多联系（一个实体可以与多个实体相关联，且另一个实体也可以与多个实体相关联））。

30. 数据库规范化是（消除数据冗余，提高数据完整性的过程，它是数据库设计的重要步骤，可以提高数据库的性能和可靠性）。 规范化的目标是（减少数据冗余，避免更新异常（插入异常、删除异常和修改异常），保证数据的完整性和一致性）。

31. 函数依赖是**（关系中属性之间的依赖关系，如果一个属性的值可以唯一确定另一个属性的值，则称存在函数依赖，例如学号可以唯一确定学生的姓名）。**

32. 完全函数依赖是**（如果一个属性集可以唯一确定另一个属性，并且该属性集中的任何一个属性都不能减少，则称存在完全函数依赖，**例如（学号，课程号）可以唯一确定学生的成绩，且学号和课程号都不能单独确定学生的成绩）。

33. 部分函数依赖是**（如果一个属性集可以唯一确定另一个属性，并且该属性集中存在可以减少的属性，则称存在部分函数依赖**，例如（学号，课程号）可以唯一确定学生的姓名，但学号可以单独确定学生的姓名）。

34. 传递函数依赖是（如果存在A->B，B->C，且B不能推出A，则称存在传递函数依赖，例如学号->系名，系名->系主任，则学号传递函数依赖于系主任）。

35. 数据库规范化的范式包括（第一范式（1NF）、第二范式（2NF）、第三范式（3NF）和BC范式（BCNF），这些范式是数据库规范化的标准，用于衡量数据库设计的质量）。

36. 第一范式（1NF）要求（**属性是原子的，不可再分，即每个属性只能包含一个值，不能包含多个值或复合值）。**

37. 第二范式（2NF）要求**（满足1NF，并且非主属性完全依赖于主键，消除部分函数依赖，即每个非主属性都必须完全依赖于主键，不能只依赖于主键的一部分）。**

38. 第三范式（3NF）要求**（满足2NF，并且非主属性不传递依赖于主键，消除传递函数依赖，即每个非主属性都必须直接依赖于主键，不能通过其他非主属性间接依赖于主键）。**

39. BC范式（BCNF）要求（满足3NF，并且每一个决定因素都包含码，消除主属性对码的部分和传递依赖，即每个属性都必须完全依赖于候选码，不能依赖于候选码的一部分或通过其他属性传递依赖于候选码）。
40. 事务是**（数据库操作的一个逻辑单元，要么全部执行成功，要么全部不执行，它是保证数据一致性的重要手段）**。事务的目的是（保证数据的一致性，即使在多个用户并发访问数据库的情况下，也能保证数据的正确性）。
41. 事务的ACID特性包括**（原子性（Atomicity）、一致性（Consistency）、隔离性（Isolation）和持久性（Durability）**，这四个特性是事务的基石，保证了事务的可靠性）。
42. 原子性（Atomicity）是指**（事务是一个不可分割的整体，要么全部执行成功，要么全部不执行，如果事务执行过程中发生错误，数据库会回滚到事务开始前的状态，保证数据的完整性）**。
43. 一致性（Consistency）是指（事务执行前后，数据库的状态必须满足预定义的约束条件，保证数据的有效性，例如银行转账，转账前后，两个账户的总金额应该保持不变）。
44. 隔离性（Isolation）是指**（多个事务并发执行时，各个事务之间互不干扰，一个事务的执行不应该影响其他事务的执行结果，保证并发访问的正确性）**。
45. 持久性（Durability）是指**（事务一旦提交，对数据库的修改是永久性的，即使系统发生故障也不会丢失，数据库会将事务的修改写入持久化存储，例如硬盘，保证数据的可靠性）**。
46. 并发控制是**（管理多个事务并发执行的技术，目的是保证事务的隔离性，防止并发操作导致的数据不一致问题，例如脏读、不可重复读和幻读）**。
47. 并发控制的主要技术包括**（封锁（Locking）、时间戳（Timestamp）和乐观锁（Optimistic Locking），这些技术可以有效地解决并发访问带来的问题）**。
48. 封锁是（一种常用的并发控制技术，通过对数据加锁，防止其他事务访问或修改该数据，保证数据的隔离性，锁可以分为共享锁和排它锁）。
49. 锁的类型包括**（排它锁（X锁，Exclusive Lock）和共享锁（S锁，Shared Lock），排它锁用于保护数据的独占访问，共享锁用于允许多个事务并发读取数据）**。
50. 排它锁（X锁）是**（一种独占锁，只允许一个事务持有，持有X锁的事务可以读取和修改数据，其他事务无法访问该数据，直到持有X锁的事务释放锁）**。
51. 共享锁（S锁）是**（一种共享锁，允许多个事务同时持有，持有S锁的事务只能读取数据，不能修改数据，其他事务也可以申请S锁，但不能申请X锁）**。
52. 死锁是**（多个事务互相等待对方释放锁，导致所有事务都无法继续执行的现象，例如事务A持有数据X的锁，等待事务B释放数据Y的锁，而事务B持有数据Y的锁，等待事务A释放数据X的锁）**。
53. 解决死锁的方法包括**（死锁预防（Prevent Deadlock）、死锁检测（Detect Deadlock）和死锁解除（Resolve Deadlock），死锁预防通过破坏死锁产生的必要条件来避免死锁，死锁检测通过检测系统中是否存在死锁来发现死锁，死锁解除通过释放部分事务的锁来解除死锁）**。
54. 数据库恢复是（在数据库发生故障后，将数据库恢复到一致状态的过程，保证数据的可靠性，故障可能包括硬件故障、软件故障和人为错误）。

55. 数据库备份是**（将数据库的数据复制到其他存储介质上，以防止数据丢失，备份可以分为全量备份、增量备份和差量备份）。**

56. 数据库恢复技术包括（**事务日志（Transaction Log）、检查点（Checkpoint）和镜像（Mirroring），这些技术可以帮助数据库从故障中恢复）。**

57. 事务日志是**（记录数据库中所有事务的执行过程，包括事务的开始、提交和修改操作，事务日志可以用于重做已提交的事务和撤销未提交的事务）。**

58. 检查点是（**数据库定期将内存中的数据刷新到磁盘上，并记录检查点信息，检查点可以缩短数据库恢复的时间）。**

59. 数据库安全性是指（保护数据库免受未经授权的访问、修改和破坏，保证数据的机密性、完整性和可用性）。

60. 数据库安全性控制包括（用户身份验证（Authentication）、访问控制（Access Control）和审计（Auditing），这些措施可以有效地保护数据库的安全）。

61. 用户身份验证是**（验证用户的身份，确认用户是否具有访问数据库的权限，常用的身份验证方式包括用户名/密码、数字证书和生物识别）。**

62. 访问控制是（控制用户对数据库对象的访问权限，例如表、视图和存储过程，访问控制可以基于角色**（Role-Based Access Control，RBAC）**或**基于权限列表（Access Control List，ACL）**）。

63. 审计是（记录用户对数据库的操作，用于追踪安全事件和进行责任追究，审计可以记录用户的登录信息、SQL语句执行情况和数据修改情况）。

64. 视图是**（一个虚拟的表，由查询定义，视图不存储实际的数据，而是通过查询从基本表中检索数据）。 视图的优点是（简化查询、提高数据安全性（可以隐藏敏感数据）、提供数据独立性）。**

65. 索引是（一种提高数据查询速度的数据结构，它可以加快数据的检索速度，类似于书籍的目录）。 索引的缺点是（占用存储空间、降低数据更新速度（因为每次更新数据都需要维护索引））。

66. 常见的索引类型包括**（B树索引（B-Tree Index）、哈希索引（Hash Index）和全文索引（Full-Text Index），不同的索引类型适用于不同的查询场景）。**

67. 存储过程是（预编译的SQL语句集合，存储在数据库中，可以被多次调用）。 存储过程的优点是（提高性能（因为存储过程是预编译的）、简化应用开发、提高安全性（可以控制存储过程的访问权限））。

68. 触发器是（与表关联的特殊存储过程，当表发生特定事件时自动执行，例如插入、更新或删除操作）。 触发器可以（实现复杂的业务规则、维护数据完整性、审计数据变化）。

69. 数据仓库是（**面向主题的、集成的、非易失的和时变的数据集合，用于支持决策分析，数据仓库的数据通常从多个数据源抽取而来，经过清洗、转换和加载后存储在数据仓库中）。**

70. 数据挖掘是（从大量数据中发现有用的模式和知识的过程，数据挖掘可以帮助企业发现潜在的商机、优化业务流程和提高客户满意度）。

71. 关系数据库管理系统（RDBMS）的例子包括（**MySQL、Oracle、SQL Server和PostgreSQL，这些RDBMS都是基于关系模型的数据库系统**）。
72. NoSQL数据库是（非关系型数据库，适用于存储非结构化和半结构化数据，例如文档数据库、键值数据库、列式数据库和图形数据库）。
73. 数据库连接池是（**管理数据库连接的组件，可以提高数据库访问性能，数据库连接池可以减少数据库连接的创建和销毁次数，提高连接的复用率**）。
74. ORM（对象关系映射）是（**将对象模型转换为关系模型的工具，简化数据库操作，ORM可以减少开发人员编写SQL语句的工作量，提高开发效率**）。
75. 数据库集群是（将多个数据库服务器组合在一起，提高数据库的可用性和性能，数据库集群可以实现负载均衡和故障转移）。
76. 分布式数据库是（**将数据库的数据分布在多个物理节点上，提高数据库的扩展性和可靠性，分布式数据库可以处理海量数据和高并发访问**）。
77. 数据加密是（**将数据转换为不可读的形式，保护数据的机密性，数据加密可以防止未经授权的用户访问敏感数据**）。
78. 数据脱敏是（**将敏感数据进行处理，使其无法识别原始数据，保护用户的隐私，数据脱敏可以用于测试环境和数据分析**）。
79. SQL注入是（一种常见的数据库安全漏洞，通过在SQL语句中插入恶意代码，获取或修改数据库信息，防止SQL注入需要对用户输入进行严格的验证和过滤）。
80. 数据库性能优化包括（**SQL语句优化、索引优化和数据库配置优化，数据库性能优化可以提高数据库的响应速度和吞吐量**）。
81. 事务隔离级别包括（**读未提交（Read Uncommitted）**、**读已提交（Read Committed）**、**可重复读（Repeatable Read）**和**串行化（Serializable）**，不同的隔离级别可以防止不同的并发问题，但也会影响数据库的性能）。
82. 数据库备份策略包括（**全量备份、增量备份和差量备份，不同的备份策略适用于不同的场景，需要根据数据的变化频率和恢复时间要求选择合适的备份策略**）。
83. 数据库监控是（**实时监控数据库的性能指标，例如CPU使用率、内存使用率、磁盘IO和SQL语句执行时间，数据库监控可以帮助管理员及时发现和解决性能问题**）。
84. 数据库调优是（**通过调整数据库的配置参数和SQL语句，提高数据库的性能，数据库调优需要根据具体的应用场景和性能瓶颈进行分析和调整**）。
85. 数据库安全审计是（**定期审查数据库的安全配置和访问日志，发现潜在的安全风险，数据库安全审计可以帮助企业提高数据库的安全水平**）。
86. 数据库高可用性是（保证数据库在发生故障时能够快速恢复，并持续提供服务，数据库高可用性可以通过数据库集群、数据复制和故障转移等技术实现）。
87. 数据库容灾是（**在发生重大灾难时，能够将数据库系统快速切换到备用站点，保证数据的安全和业务的连续性，数据库容灾需要建立完善的备份和恢复机制**）。

88. 数据库云化是**（将数据库系统部署在云平台上，利用云计算的弹性伸缩和高可用性，降低数据库的运维成本和提高数据库的可靠性）。**

89. 数据库自治是（利用人工智能技术，实现数据库的自动化管理和优化，例如自动索引推荐、自动SQL调优和自动故障诊断）。

90. 数据湖是**（存储各种类型和格式的数据的存储库，包括结构化数据、半结构化数据和非结构化数据，数据湖可以用于数据分析、数据挖掘和机器学习）。**

91. ETL（抽取、转换和加载）是（将数据从不同的数据源抽取出来，经过清洗、转换和加载后存储到数据仓库中的过程，ETL是构建数据仓库的关键步骤）。

92. 数据治理是**（管理组织的数据资产，确保数据的质量、安全和合规性，数据治理包括数据标准、数据质量、数据安全和数据合规等方面）。**

93. 元数据管理是**（管理描述数据的数据，例如数据的名称、类型、大小、创建时间等，元数据管理可以帮助用户更好地理解和使用数据）。**

94. 数据血缘分析是**（追踪数据的来源和转换过程，了解数据的生命周期，数据血缘分析可以帮助用户理解数据的含义和评估数据的质量）。**

95. 数据库建模工具是**（用于设计数据库结构的工具，例如ERwin、PowerDesigner和Visio，数据库建模工具可以帮助用户可视化数据库结构和提高设计效率）。**

96. SQL优化工具是**（用于分析和优化SQL语句的工具，例如SQL Developer、Toad和DataGrip，SQL优化工具可以帮助用户发现SQL语句的性能瓶颈并提供优化建议）。**

97. 数据库迁移工具是（用于将数据库从一个平台迁移到另一个平台的工具，例如SQL Server Migration Assistant、Oracle SQL Developer和pg_dump/pg_restore，数据库迁移工具可以简化数据库迁移的过程）。

98. 数据库测试工具是**（用于测试数据库的性能、安全性和可靠性的工具，例如JMeter、LoadRunner和SQLMap，数据库测试工具可以帮助用户发现数据库的潜在问题）。**

99. 数据库文档是（描述数据库结构、功能和使用方法的文档，数据库文档可以帮助用户更好地理解和使用数据库）。

100. 数据库社区是**（一个由数据库用户、开发者和专家组成的社群，可以在社区中交流经验、分享知识和解决问题，常见的数据库社区包括Stack Overflow、CSDN和GitHub）。**

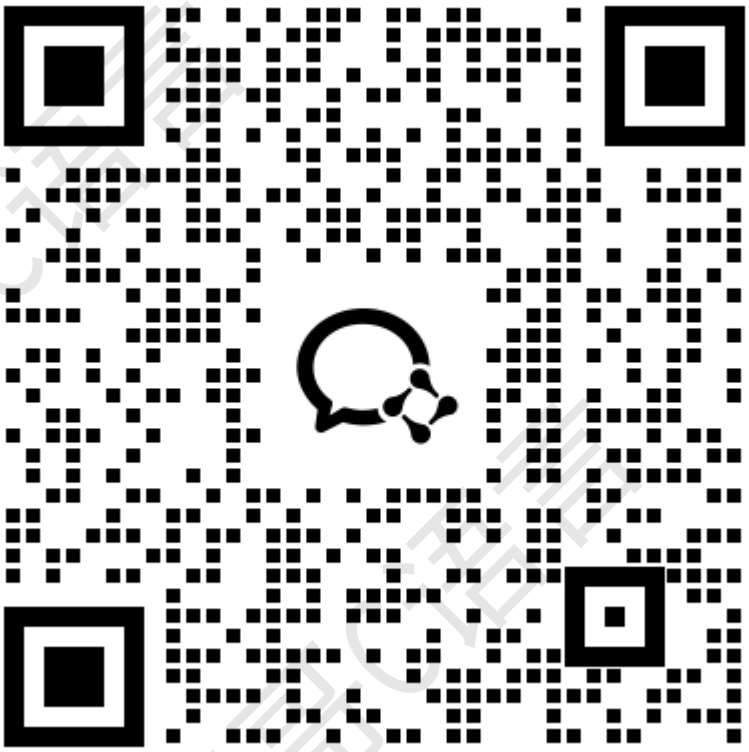
101. 数据库发展趋势是**（云数据库、NoSQL数据库、NewSQL数据库、图数据库、时序数据库和多模数据库，这些新型数据库可以满足不同应用场景的需求）。**

102. 数据库学习资源包括**（官方文档、在线课程、书籍和博客，这些资源可以帮助用户系统地学习数据库知识）。**

103. 数据库认证是**（证明个人具备数据库相关技能和知识的证书，例如Oracle Certified Professional、Microsoft Certified Database Administrator和MySQL Database Administrator，数据库认证可以提高个人的职业竞争力）。**

104. 数据库面试准备包括**（掌握数据库基本概念、SQL语句、数据库设计和性能优化等知识，并准备常见的面试问题，充分的准备可以提高面试的成功率）。**

105. 数据库职业发展方向包括**（数据库管理员、数据库开发工程师、数据仓库工程师、数据分析师和数据科学家，不同的职业方向需要不同的技能和知识）。**



七、软件工程 专升本高频考点背诵（极细化补充版）

（共99个高频考点+考前速记）

1. 软件工程是**（运用系统化的、规范的、可量化的方法来开发、运行和维护软件的工程学科）。**核心目标是（在预算内、按时交付满足用户需求的高质量软件产品）。

2. 软件生命周期是**（软件产品从构思、开发、使用到最终退役的完整过程）。**典型阶段包括（需求分析、设计、编码、测试、部署和维护）。

3. 需求分析是（明确软件系统需要做什么，解决“做什么”的问题）。 关键任务是**（获取、分析、验证和记录用户需求，形成需求规格说明书）。**

4. 需求获取方法包括（用户访谈、问卷调查、原型法、用例建模、联合需求计划（JRP）等）。 选择合适的方法（取决于项目规模、用户参与度和需求复杂性）。
5. 软件设计是**（将需求分析的结果转化为具体的实现方案，解决“怎么做”的问题）。** 主要分为（概要设计（系统架构设计）和详细设计（模块、算法设计））。
6. 概要设计（又称系统设计或高层设计）确定（**系统的总体结构、模块划分、模块接口、数据结构和数据库设计等）。** 目标是**（建立清晰、可维护的系统架构）。**
7. 详细设计（又称模块设计或底层设计）描述（**每个模块的内部实现细节，包括算法、数据结构、接口规范等）。** 目标是**（为编码提供详细的指导）。**
8. 软件编码是（将设计方案转化为可执行的程序代码）。 强调（遵循编码规范、编写清晰易懂、可维护的代码）。
9. 软件测试是（发现软件缺陷、评估软件质量的过程）。 目标是**（尽早发现并修复缺陷，确保软件满足用户需求）。**
10. 软件测试类型按阶段划分包括**（单元测试、集成测试、系统测试和验收测试）。** 不同阶段（侧重点不同，需要采用不同的测试策略和方法）。
11. 单元测试是**（对软件中的最小可测试单元（通常是函数或方法）进行测试）。** 重点是（验证代码的逻辑正确性、边界条件和异常处理）。
12. 集成测试是**（将多个单元组合在一起进行测试，验证模块之间的接口和交互是否正确）。** 常见的集成测试策略包括（自顶向下、自底向上和混合式）。
13. 系统测试是**（对整个软件系统进行全面的测试，验证系统是否满足所有的需求）。** 常见的系统测试类型包括（功能测试、性能测试、安全测试、兼容性测试等）。
14. 验收测试是**（由用户或客户进行的测试，验证软件是否满足用户的业务需求和期望）。** 验收测试通过（标志着软件开发过程的结束）。
15. 软件维护是**（在软件交付后，修改、完善和更新软件系统的过程）。** 目的是（修复缺陷、适应环境变化、增加新功能和提高性能）。
16. 软件维护类型包括（**纠错性维护（修复缺陷）**、**适应性维护（适应环境变化）**、**完善性维护（增加新功能）**和**预防性维护（提高可维护性）**）。
17. 软件质量是**（软件产品满足用户需求的程度，是软件工程的核心目标）。** 软件质量特性包括（功能性、可靠性、易用性、效率、可维护性、可移植性等）。
18. 软件质量保证（SQA）是**（一系列活动，旨在确保软件产品满足预定的质量标准和要求）。** SQA活动包括（评审、测试、审计、配置管理等）。
19. 软件配置管理（SCM）是（管理软件配置项（例如源代码、文档、测试用例）的过程）。 SCM目标是（控制变更、保证一致性、支持版本管理和构建管理）。
20. 软件过程是**（开发和维护软件的一系列活动、方法和实践）。** 软件过程模型（描述了软件开发的各个阶段和活动之间的关系）。

21. 瀑布模型是**（一种线性的、顺序的软件过程模型，每个阶段严格按照顺序执行，前一个阶段完成后才能进入下一个阶段）。**优点是（简单、易于管理），缺点是（灵活性差、难以适应需求变化）。
22. 迭代模型是**（一种循环的软件过程模型，将软件开发过程分为多个迭代周期，每个迭代周期都包含需求分析、设计、编码、测试等阶段）。**优点是（可以尽早发现和解决问题、适应需求变化），缺点是（需要进行迭代规划和管理）。
23. 增量模型是**（一种将软件系统分解为多个增量构建的软件过程模型，每个增量构建都提供一部分功能）。**优点是（可以尽早交付可用的软件系统、降低开发风险），缺点是（需要进行增量规划和管理）。
24. 螺旋模型是**（一种风险驱动的软件过程模型，将软件开发过程分为多个螺旋周期，每个螺旋周期都包含需求分析、风险评估、设计、编码、测试等阶段）。**优点是（可以有效地管理风险、适应需求变化），缺点是（需要进行风险评估和管理）。
25. 敏捷开发是**（一种轻量级的、迭代的软件开发方法，强调快速迭代、持续交付和团队协作）。**敏捷方法包括（Scrum、极限编程（XP）、看板（Kanban）等）。
26. Scrum是一种**（迭代增量的敏捷软件开发框架，强调短周期迭代（Sprint）、每日站会（Daily Scrum）和回顾会议（Sprint Retrospective））。**
27. 极限编程（XP）是一种**（强调代码质量、测试驱动开发、结对编程和持续集成的敏捷软件开发方法）。**
28. 看板（Kanban）是一种**（强调可视化工作流程、限制在制品数量和持续改进的敏捷软件开发方法）。**
29. UML（统一建模语言）是**（一种用于描述、可视化、构建和文档化软件系统的标准建模语言）。**UML图包括（用例图、类图、时序图、状态图、活动图、组件图和部署图）。
30. 用例图**（描述系统的功能需求，由用例、参与者和关系组成）。**用于（获取和表达用户需求）。
31. 类图**（描述系统的静态结构，由类、属性和方法组成）。**用于（设计和表达系统的对象模型）。
32. 时序图**（描述对象之间的交互，强调时间顺序）。**用于（分析和设计对象之间的协作）。
33. 状态图**（描述对象的状态变化）。**用于（分析和设计对象的状态机）。
34. 活动图**（描述活动的流程）。**用于（分析和设计业务流程和工作流程）。
35. 组件图**（描述系统的组件及其关系）。**用于（设计和表达系统的物理结构）。
36. 部署图**（描述软件系统在硬件上的部署）。**用于（规划和表达系统的部署架构）。
37. 软件项目管理是**（规划、组织、领导和控制软件项目的过程，确保项目在预算内、按时交付高质量的软件产品）。**
38. 软件项目估算包括**（工作量估算、成本估算和进度估算，是项目规划的重要组成部分）。**准确的估算（是项目成功的关键）。

39. 软件风险管理是**（识别、评估和控制软件项目风险的过程，降低风险对项目的影响）。**风险管理（贯穿于整个项目生命周期）。
40. 软件质量度量是**（量化软件质量的过程，通过度量指标来评估软件的质量特性）。**质量度量（为质量改进提供依据）。
41. 软件复用是**（重用已有的软件组件或代码，以减少开发工作量和提高软件质量）。**复用（可以提高开发效率和降低成本）。
42. 软件体系结构是**（软件系统的基本结构，包括组件、关系和约束，是软件设计的蓝图）。**体系结构（影响系统的可维护性、可扩展性和性能）。
43. 设计模式是**（解决软件设计中常见问题的可重用解决方案，是经验总结）。**掌握设计模式（可以提高软件设计的水平）。
44. 软件测试用例是**（用于测试软件系统的输入、预期输出和执行条件，是测试的基础）。**编写高质量的测试用例（可以有效地发现软件缺陷）。
45. 软件测试覆盖率是**（衡量测试用例覆盖软件代码的程度，评估测试的充分性）。**提高测试覆盖率（可以提高软件质量）。
46. 软件缺陷报告是**（描述软件缺陷的文档，包括缺陷描述、重现步骤、严重程度和优先级，是缺陷跟踪和修复的基础）。**
47. 软件版本控制是**（管理软件版本的过程，包括版本创建、版本合并和版本发布，是配置管理的重要组成部分）。**
48. 常用的版本控制工具包括（**Git（分布式版本控制系统）**、**SVN（集中式版本控制系统）**和**CVS（较早的版本控制系统）**）。
49. 软件配置管理工具是**（用于管理软件配置项的工具，包括版本控制、构建管理和发布管理，提高配置管理的效率）。**
50. 持续集成（CI）是**（一种软件开发实践，将代码频繁地集成到共享仓库中，并进行自动化构建和测试，尽早发现集成问题）。**
51. 持续交付（CD）是**（一种软件开发实践，将软件自动地发布到生产环境中，缩短发布周期）。**
52. DevOps是**（一种强调开发、运维和安全协作的软件开发文化，提高软件交付的速度和质量）。**
53. 微服务架构是**（一种将应用程序分解为小型、自治的服务的架构风格，提高系统的可扩展性和可维护性）。**
54. 容器化技术是**（一种将应用程序及其依赖项打包到容器中的技术，例如Docker，提高应用程序的可移植性和部署效率）。**
55. 软件安全是**（保护软件系统免受恶意攻击和未经授权访问的过程，保证数据的机密性、完整性和可用性）。**
56. 常见的软件安全漏洞包括（**SQL注入**、**跨站脚本攻击（XSS）**和**缓冲区溢出**，需要采取相应的安全措施）。

57. 软件安全测试是（发现软件安全漏洞的过程，常用的方法包括**渗透测试**、**代码审计和漏洞扫描**）。 软件安全测试（是保证软件安全的重要手段）。

58. 静态代码分析是（**在不运行代码的情况下，检查代码中的潜在问题**，例如安全漏洞、代码风格问题和潜在的错误）。 静态代码分析工具（可以自动化地进行代码检查）。

59. 动态代码分析是（**在运行代码的情况下，检查代码中的潜在问题**，例如内存泄漏、性能瓶颈和安全漏洞）。 动态代码分析（需要运行代码，可以发现一些静态分析无法发现的问题）。

60. 软件法律法规是**（规范软件开发和使用的法律法规，例如著作权法、专利法和合同法）**。 了解软件法律法规（可以避免法律风险）。

61. 软件伦理是**（指导软件开发人员行为的道德准则，例如保护用户隐私、尊重知识产权和避免损害社会利益）。** 遵守软件伦理（是软件工程师的责任）。

62. 软件工程职业道德是（指导软件工程师行为的职业道德准则，例如诚实守信、精益求精和勇于承担责任）。 良好的职业道德（是软件工程师的立身之本）。

63. 软件工程教育是**（培养合格的软件工程师的教育体系，包括理论知识、实践技能和职业素养）。** 高质量的软件工程教育（是软件产业发展的基石）。

64. 软件工程研究是**（探索软件工程领域的新技术和新方法，以提高软件开发的效率和质量）。** 软件工程研究（推动软件工程领域的进步）。

65. 软件工程标准化是**（制定和推广软件工程领域的标准，以提高软件开发的规范性和互操作性）。** 软件工程标准（提高软件开发的效率和质量）。

66. 软件工程国际化是**（使软件系统能够适应不同国家和地区的语言、文化和法律法规）。** 软件国际化（是软件走向全球市场的必要条件）。

67. 软件工程本地化是**（将软件系统翻译和调整到特定国家和地区的语言和文化）。** 软件本地化（提高用户体验）。

68. 软件工程全球化是**（在全球范围内进行软件开发和销售）。** 软件全球化（是软件产业发展的趋势）。

69. 软件工程外包是**（将软件开发任务委托给外部公司或团队）。** 软件外包（可以降低开发成本）。

70. 软件工程众包是**（将软件开发任务委托给大量分散的个人或团队）。** 软件众包（可以利用社会力量解决问题）。

71. 软件工程开源是**（将软件源代码公开，允许用户自由使用、修改和分发）。** 开源软件（促进技术创新和知识共享）。

72. 软件工程商业模式是**（软件企业如何创造价值和获取利润的模式）。** 商业模式（是软件企业生存和发展的关键）。

73. 软件工程创新是**（在软件工程领域创造新的技术、方法和商业模式）。** 创新（是软件产业发展的动力）。

74. 软件工程未来发展趋势是**（人工智能、大数据、云计算、物联网和区块链）。** 关注未来发展趋势（有助于把握软件工程的未来）。

75. 软件工程与人工智能的结合是**（利用人工智能技术提高软件开发的效率和质量，例如自动化代码生成、自动化测试和智能缺陷预测）。**

76. 软件工程与大数据的结合是**（利用大数据技术分析软件开发过程中的数据，以提高软件开发的效率和质量，例如代码质量分析、缺陷预测和用户行为分析）。**

77. 软件工程与云计算的结合是**（将软件开发工具和平台部署在云端，以提高软件开发的灵活性和可扩展性）。**

78. 软件工程与物联网的结合是**（开发物联网设备和应用的软件系统，例如智能家居、智能交通和智能医疗）。**

79. 软件工程与区块链的结合是**（利用区块链技术构建安全可靠的软件系统，例如数字身份认证、供应链管理和版权保护）。**

80. 软件工程的挑战是**（需求变化快、技术更新快、人员流动性大和安全威胁日益严重）。** 应对挑战（需要不断学习和创新）。

81. 软件工程的机遇是（人工智能、大数据、云计算、物联网和区块链等新技术的发展，以及全球化和开源软件的普及）。 抓住机遇（可以实现个人和企业的发展）。

82. 软件工程师的素质要求是**（扎实的理论基础、熟练的编程技能、良好的沟通能力、团队合作精神和持续学习能力）。** 提高自身素质**（是成为优秀软件工程师的必要条件）。**

83. 软件工程师的职业发展路径是（初级软件工程师、中级软件工程师、高级软件工程师、架构师和技术经理）。 规划职业发展（有助于实现职业目标）。

84. 软件工程的价值是**（提高软件开发的效率和质量，降低软件开发的成本和风险，并为用户提供更好的软件产品和服务）。**

85. 软件工程的意义是**（推动社会进步，促进经济发展和改善人们的生活）。**

86. 软件工程的使命是**（构建可靠、安全、高效和易用的软件系统，为人类创造更美好的未来）。**

87. 软件工程的愿景是**（成为引领科技进步和社会发展的核心力量）。**

88. 软件工程的口号是**（精益求精，追求卓越，创新驱动，服务社会）。**

89. 软件工程的座右铭是**（代码改变世界，技术成就梦想）。**

90. 软件工程的未来是**（充满挑战，充满机遇，充满希望）。**

91. 软件度量是**（对软件开发过程和产品进行量化分析，为决策提供依据）。** 常用的度量包括（代码行数、功能点、缺陷密度等）。

92. 软件过程改进是**（通过分析和评估现有过程，识别改进机会并实施改进措施，以提高软件开发效率和质量）。** 常用的改进模型包括（CMMI）。

93. 需求管理是**（对需求进行收集、分析、验证、确认和变更控制的过程，确保需求的一致性和完整性）。** 需求管理工具（例如JIRA）。

94. 软件维护成本是**（软件生命周期中重要的组成部分，通常高于开发成本）。** 降低维护成本（是软件工程的重要目标）。

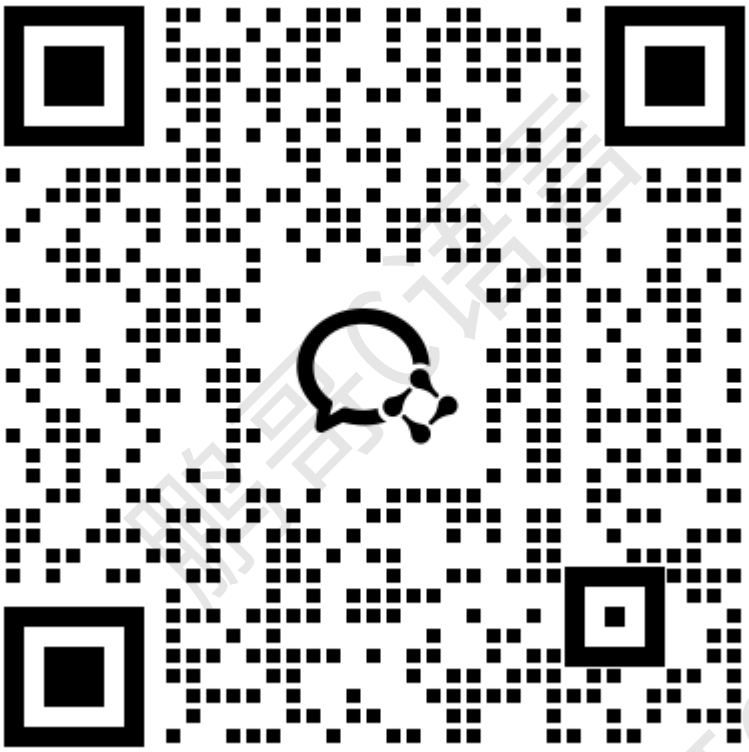
95. 遗留系统是**（已经存在并正在使用的软件系统，通常年代久远，技术陈旧）。**维护遗留系统（面临诸多挑战）。

96. 软件再工程是（对遗留系统进行分析、改造和升级，以提高其可维护性、可扩展性和性能）。

97. 软件逆向工程是**（从可执行程序或二进制代码中提取设计信息和代码的过程）。**用于**（理解和分析软件系统）。**

98. 软件项目风险评估是**（识别和评估项目潜在风险的过程，为风险应对提供依据）。**常用的风险评估方法（德尔菲法、头脑风暴法）。

99. 软件质量保证体系是**（组织为了保证软件质量而建立的一套制度、流程和方法）。**常用的质量保证标准**（ISO 9000）**。



八、计算机基础知识

（共137个高频要点+速记）

1. 世界上第一台计算机于 **1946** 年诞生，它的名字叫 **ENIAC.**

2. 在计算机运行中，把程序和数据一样存放在内存中，提出并论证 这个理论的研究小组领导是**冯诺依曼 .**
3. 语言处理程序发展经历的前三个阶段是：**机器语言、汇编语言、高 级语言.**
4. 计算机内存容量的基本单位是**字节 .**（字节 (Byte) 是常用的基本单位。 更小的单位是位 (bit)。
5. 微处理器研制成功的时间是：**1971年.**
6. 微型计算机发展的标志是**微处理器.**
7. 计算机发展阶段的划分标志为：**物理器件**
8. 世界上第一台电子计算机所采用的逻辑元件是**电子管**
9. 使用超大规模集成电路制造的计算机应该归属 **第四代**
10. 微型计算机的问世，主要是由于出现了 **超大规模集成电路**
11. 按计算机应用的分类，办公自动化属于 **数据处理**
12. 银行利用计算机进行存贷款业务管理属于计算机应用领域的**数据处理**
13. 财务管理，情报检索，库存管理等属于计算机应用领域的**数据处理**
14. CAD 是计算机主要应用领域之一，其含义是**计算机辅助设计 .**（CAD 主要用于工程设计、建筑设计、产品设计等领域。）
15. 微型计算机中使用数据库管理系统，是计算机应用中的**信息管理**
16. CAI 是计算机应用的一个重要领域，它的含义是**计算机辅助教学**
17. 应用计算机最早的领域是**科学计算**
18. 实现现代化工业生产过程自动化的主要手段是**用计算机进行实时控制**
19. 个人计算机属于**微型计算机**
20. 中国国防科技大学研制的“**银河**”计算机属于**巨型计算机**
21. 实现计算机网络的最大好处是**资源共享**
22. 计算机系统是指**硬件和软件系统**
23. 在计算机系统中，通常所说的计算机系统自愿指的是**硬件、软件，数据 .**
24. 所谓计算机的“裸机”是指**不装备人和软件的计算机**
25. 微型计算机的基本组成是**微处理器，存储器，输入输出设备**
26. 微型计算机的主机包括 **CPU 和内存储器**
27. 运算器和控制器的总称是 **CPU**
28. 微处理器又称**中央处理器**
29. 计算机的核心部件是**中央处理器**
30. 8088，80286，80386，80486 指的是不同型号的**（微处理器（或者中央处理器在或者 CPU））**

31. 算术逻辑单元（简称 ALU）主要提供**算术运算和（逻辑运算）**
32. CPU 中控制器的主要功能是**（识别指令和控制指令的执行）**
33. 微型计算机中，控制器的基本功能是**（控制系统各部件正确地执行程序）**
34. 在计算机系统中，指挥，协调计算机工作的设备是**（控制器）**
35. 微型计算机中的 I/O 接口卡位于**（总线与外设之间）**
36. I/O 设备的含义是**（输入/输出设备）**
37. 下列部件中，能直接与 CPU 相连接的是**（内存储器）**
38. 在微型计算机中，硬盘连同其驱动器属于**（外(辅助)存储器）**
39. 下列设备中，即可作为输入设备，又可作为输出设备的是**（磁盘驱动器也就是软盘驱动器）**
40. 输入设备：**键盘，鼠标，光笔，扫描仪**
41. 输出设备：**显示器，打印机，绘图仪，音箱**
42. 外部设备：**输出输入设备，辅助存储器**
43. 计算机中的 CRT 是指**（阴极射线显示器）**
44. 以 SVGA,EGA,VGA 标志着不同规格和性能的设备是**（显示器）**
45. 作为显示器主要参数之一的分辨率，其含义是**（显示屏幕上光栅的列数和行数）**
46. 下面叙述中有错误的一条是**（显示器的分辨率与微处理器的型号有关）**
47. 属于击打式打印机的是**（针式打印机）**
48. 打印效果最佳的一种是**（激光打印机）**
49. 一组连接计算机各部件的公共通信线称为总线，组成是**（地址线，数据线和控制线）**
50. 鼠标器通常连接在**（串行接口上）**
51. 具备即插即用功能的接口是 **USB**
52. 主机板上 CMOS 芯片的主要用途是**（存储时间，日期，硬盘参数与计算机配置信息）**
53. 在计算机系统中，软件指的是**（程序，数据及其有关的文档资料）**
54. 软件与程序区别是**（软件是程序及开发使用和维护所需要的所有文档的总称，而程序是软件的一部分）**
55. 计算机的软件系统分为**（系统软件和应用软件）**
56. 系统软件与应用软件的相互关系是**（后者以前者为基础）**
57. 应用软件是**（用于各领域的专用软件）**
58. “最靠近”计算机硬件的是**（操作系统）**
59. 系统软件中的核心部分是**（操作系统）**
60. 引入操作系统的主要目的是方便用户及**（提高软，硬件资源的利用率）**

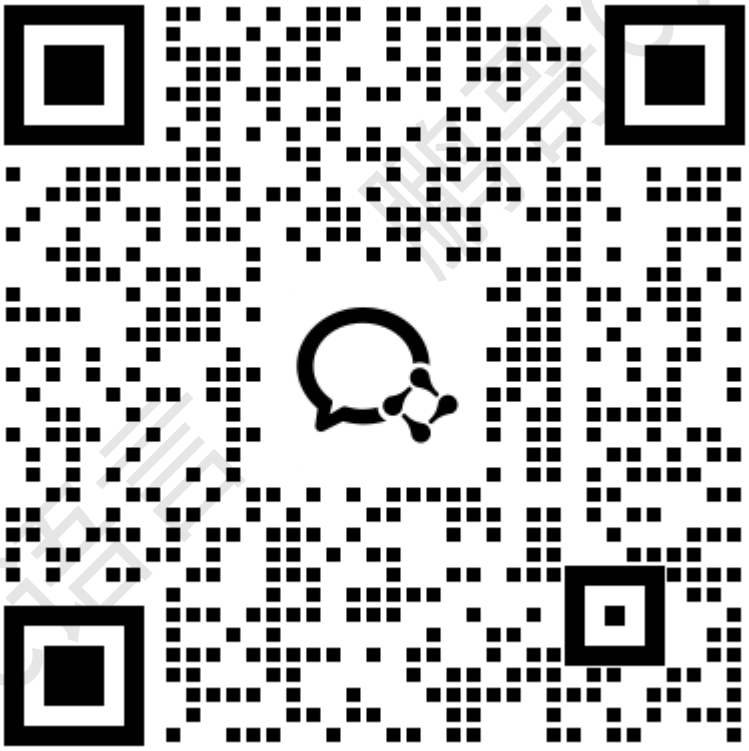
61. 操作系统的主要作用不包括**（预防和消除计算机病毒的侵害）**
62. 操作系统有：**MS-DOS，UNIX，Windows**
63. 计算机所能识别的一组不同指令的集合称为**（指令系统）**
64. 系统软件有：**编译程序，操作系统，数据库管理系统**（注意：C 语言源程序不是系统软件）
65. SQL Server,Access,Foxpro 被称为**（数据库管理系统）**
66. 用于规定计算机执行的操作及操作数地址的一个二进制位串称为**（指令）**
67. 某学校的工资管理程序属于**（应用程序）**
68. 完成一步基本运算或判断，需要计算机的 CPU 执行一个**（指令）**
69. 计算机能直接执行的程序是**（机器语言程序）**
70. 由二进制编码构成的语言是**（机器语言）**
71. 机器指令是二进制代码，能被计算机**（直接执行）**
72. 汇编语言是一种**（面向机器的低级符号语言）**
73. 机器语言和汇编语言都是面向**（机器）的语言**
74. 通常，人们把用高级语言编写的程序称为**（源程序）**
75. 能将高级语言源程序转换成目标程序的是**（编译程序）**
76. 用 C 语言编译的源程序，要变为目标程序，必须经过**（编译）过程**
77. 编译程序产生**目标程序**
78. 最适合信息管理的计算机语言是**（数据库语言）**
79. BASIC 语言是一种**（高级语言）**
80. 属于面向对象的程序设计语言有**（Visual Basic）**
81. 某计算机的存储器容量是 4MB，它是**2 的（22）次方**

4 是 2 的 2 次方，1024 是 2 的 10 次方，故便是 10+10+2=22

82. 存储容量单位：**1Byte=8bit, 1024B=1KB, 1024KB=1MB, 1024MB=1GB**
83. 计算机中的地址是**（存储单元的有序编号）**
84. 给微型计算机的内存储器进行编址的单位是**（字节）**
85. 计算机中最小的数据单位是**（位）**
86. bit 的意思是**（二进制位）**
87. Byte 中文意思是（字节）
88. 计算机中组成一个字的位数叫做该字的**（字长）**
89. CPU 处理的数据基本单位是**字**，一个字的字长**（与 CPU 芯片的型号有关）**
90. 80486 微型计算机的字长是**（32 位）**

91. 假设一台计算机的字长是 4 个字节，这意味着**（在 CPU 中作为一个整体加以传送处理的二进制代码为 32 位）**。原因：4Byte=4x8=32 个二进制位
92. 32 位微型计算机中的 32 是指该微型计算机**（能同时处理 32 位二进制数）**
93. 指令的组成部分包括**操作数地址和（操作码）**
94. 8MB 表示为 **8MB（兆字节）**
95. “Pentium II /350” “ PentiumIII/450”中的“350” “450”的含义是**（CPU 时钟频率）**
96. 微型计算机的性能主要取决于**（中央处理器）**
97. 内存储器与 CPU 直接交换信息，与外存储器相比存取速度更快，但价格也更贵。
98. CPU 不能直接访问的存储器是**（CD-ROM）**
99. 在计算机中能直接与 CPU 交换数据的是**（高速缓冲和主存储器）**
100. 内存储器和外存储器相比较，内存储器的主要特征是**（能存储正在运行的程序）**
101. 在微型计算机中 **RAM** 的功能是**（存放可读写的程序和数据）**
102. SRAM 存储器是**（静态随机存储器）**
103. 在半导体存储器中，动态随机存储器 DRAM 的特点是**（每隔一定的时间刷新一次）**
104. 在微型计算机系统中，最基本的输入/输出模块 BIOS 存放在**（ROM）**中
105. **ROM** 的中文名称是**（只读存储器）**
106. 在一台计算机上 **ROM BIOS** 的内容**（固定不变）**
107. RAM 特点是**（可随机读写数据，断点后数据将全部丢失）**
108. 存储系统中的 PROM 是指**（可编程只读存储器）**
109. 与微型计算机的辅助存储器直接进行数据传送的部件是**（内存储器）**
110. 通常说的内存条是指**（RAM）**
111. 使用 Cache 可以提高计算机运行速度，这是因为**（Cache 缩短了 CPU 的等待时间）**
112. 计算机多层次的存储体系结构包括**（主存储器，辅助存储器和高速缓冲存储器）**
113. 磁盘缓冲区位于**（主存储器内）**
114. 磁盘存储器存，取信息的最基本单位是**（扇区）**
115. 3.5 英寸软盘片如果移动一个角上的滑块露出一个小孔，则**（只能读不能写）**
116. 一张 3.5 英寸软盘片双面高密度软盘片的容量为**（1.44MB）**
117. （720KB3.5 英寸）磁盘可以在 **1.44MB3.5 英寸软盘驱动器** 中使用
118. 计算机中的磁盘驱动器指示灯亮时，**（不能打开该驱动器开关和关闭主机电源）**
119. 磁盘的磁道是同心圆，而 CD-ROM 光盘上记录信息的光道是**（螺旋线）**
120. 微型计算机的硬盘是该机的**（外(辅)存储器）**

121. 对多媒体计算机的正确理解是**（能综合处理文字，图形，影响连接与声音等信息的计算机）**
122. 在计算机内，多媒体数据最终存在的形式是**（二进制代码）**
123. 多媒体计算机系统包括多媒体计算机软件系统和**（多媒体计算机硬件系统）**
124. 在多媒体使用中，一般背景音乐用**（WAV）**文件，解说用**（MIDI）**文件。
125. 计算机中的信息用二进制表示的主要理由是**（元器件性能所致）**
126. 在计算机内部表示数据的进制是**（二进制）**
127. 使用数字波形法表示声音信息时，采样频率越高，则数据量**（越大）**
128. 准确地描述字符的 ASCII 编码在机器中表示方法的应是**（使用8 位二进制代码，最左边一位为0）**
129. 关于字符之间 ASCII 码值大小关系**（d>D>空格符）**
130. 在一个非零的无符号二进制整数后加两个零得到一新数，该新数是原数的**（四倍）**
131. 设一张软盘已染上病毒，能清除病毒的措施是**（格式化软盘）**，磁盘也是一样
132. 防止软盘感染病毒的有效方法是**（对软盘进行写保护）**
133. 微处理器又称为中央处理器，它的组成包括**（运算器，控制器，寄存器，计数器）**
134. 二进制的优越性包括**（可行性，简易性，逻辑性，可靠性）**
135. 计算机能直接执行的指令包括两部分，它们是**（操作码，操作数）**
136. 在计算机工作时，内存储器用来存储**（运算所需的程序，运算所需的数据）**
137. **JPEG** 压缩**静止图像**， **MPEG** 压缩**运动图形**

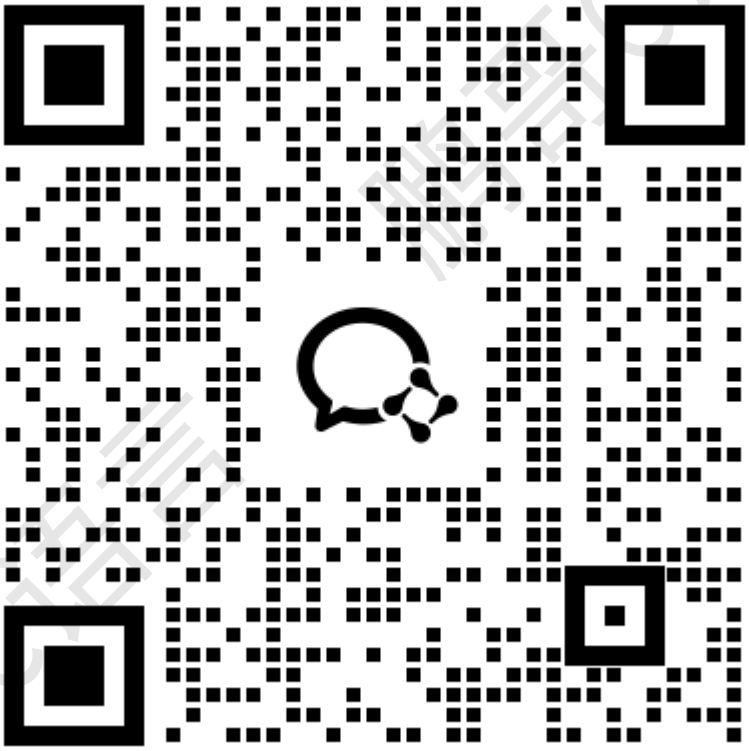


九、Windows2000 系统使用

（共26个高频考点+考前速记）

1. 操作系统的主体是**（程序）**.（操作系统是由一系列程序组成的。）
2. 在操作系统中，文件系统的主要作用是**（实现对文件的按名存取）**.（文件系统负责管理磁盘空间，组织文件和目录，并提供按名称访问文件的机制。）
3. Windows2000 是一种**（多任务图形方式操作系统，独立于 dos 的32 位操作系统，特点是独立的操作系统，能直接启动）**
4. Windows2000 的菜单命令后面有省略号“…”，就表示系统在执行此菜单命令时，为获取更多的信息需要打开**（对话框）**
5. 在 Windows2000 中，浏览系统资源可以通过**我的电脑**和**资源管理器**
6. 在 Windows2000 桌面的“任务栏”中，显示的是**（所有已打开的窗口的图标）**
7. 在 Windows2000 中，“任务栏”**（即可改变位置也能改变大小）**，基本作用是**（实现窗口之间的切换）**
8. 当运行的应用程序最小化后，该应用程序的状态是**（后台运行）**

9. 若 Windows 的某个应用程序的窗口处于最大化，双击其标题栏的作用等价于单击窗口的（“还原”按钮）
10. 要查找所有的 BMP 格式文件，应在“搜索”对话框“要搜索的文件或文件夹名”中输入的名称是（*.BMP）其他格式的文件也是一样的。
11. 在 Windows2000 中“资源管理器”窗口的文件夹前的“+”号表示（给文件夹图标含有下级文件夹 且未展开）“-”表示（不含下级文件夹）
12. 在 Windows2000 中，文件名中不能有 <>:"/\|?* ，例如 A<B.C 是不对的
13. 在 Windows2000 中，“剪贴板”是（占内存的一块区域），“回收站”则是占用（硬盘的空间）
14. 在 Windows2000 的四种文件属性中，通常不能由用户自行设置或修改的属性是（系统），“隐藏” “存档” “只读”都可以修改或设置
15. 在 Windows2000 的“资源管理器”窗口中，当选定文件夹并按了“Shift+Del”后，所选定的文件夹将（被删除但不放入“回收站”）
16. 在 Windows2000 的中文输入方式下，要输入中文标点符号“、”，应按的键是（\）
17. 实行中西文输入方式的切换，要按的键是（Ctrl+空格）
18. Windows2000 的“磁盘扫描程序”可用于检查，诊断和修复各种类型的磁盘损坏和错误。它能自动检测和修复磁盘的（逻辑错误）
19. 磁盘处于写保护的状态下（能读出，但不能删改其中的数据，也不能写入新数据）
20. 在 Windows2000 中，要设置屏幕属性和改变屏幕保护程序的设置，都可在“控制面板”下的“显示”项进行
21. 在 Windows2000 中，“记事本”放在（附件）中，用于（编辑小型文本文件）
22. 在 Windows2000 中，“画图”放在（附件）中，它的默认文件类型是（BMP）
23. 在 Windows2000 中，能够把整个屏幕复制到剪贴板上，应该按的键是（Print Screen）
24. 在 Windows2000 中，进行不同窗口之间的切换操作的组合键是（Alt+Tab）
25. 当鼠标指针为沙漏形时，表示应用程序正在运行，用户只有（等待）
26. 在 Windows2000 中，文件名可以使用的是（字母，汉字，空格，下划线）



十、Word 2000 的使用

（共37个高频要点+速记）

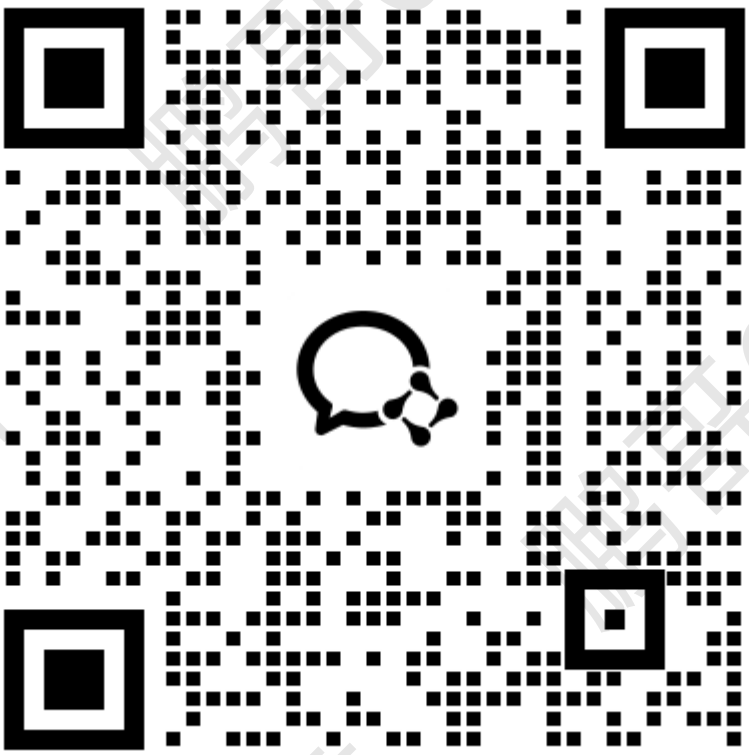
1. 在 Word 2000 执行“复制”操作后，“剪贴板”中默认保存复制的内容次数最多为**（12）**（但这个数量可以通过设置进行更改。）
2. 在选中字符后，需给该字符加上“赤水情深”的动态效果，应在“字体”对话框中选择的选项卡为**（动态效果）**
3. 想在 Word 2000 主窗口中显示段落标记，应使用菜单项为**（视图）**
4. 以下有关创建目录的说法，正确的是**（在创建目录之前，定义“样式”并对创建的内容应用“样式”）**
5. 依次打开三个 Word 2000 文档，每个文档都有修改，在修改完后为了一次性保存着学文档，正确的操作是**（按 Shift 键的同时，单击“文件”—“全部保存”命令）**
6. 使用 Word 2000 时，在输入过程中，用户输入“南”，文档中出现“南开大学”。这是利用了**（“工具”菜单中的“自动更正”功能）**
7. 在“打印”对话框的“页面范围”栏的“当前页”项是为**（插入光标所在的页）**
8. 把段落的第一行向右移动两个字符的位置，正确的选项是**（“格式”菜单中的“段落”命令）**

9. 要删除表格中的某个单元格并使右侧单元格左移，正确的操作是选择**（“表格”菜单中的“删除单元格”命令）**
10. 在 Word 2000 中，如果在“文件”菜单的子菜单下部列出一些文件名，这些文件名是**（最近在 Word2000 中打开，处理过的文档）**
11. “页眉页脚”命令所在的菜单项是**（“视图”菜单）**
12. 在 Word 2000 中，“项目符号”可以**（改变，可自动顺序生成，对于同一段落项目符号和编号不能同时存在）**
13. 在编辑文档时，需要在输入新的文字的同时替换原有文字，最快捷的操作步骤是**（选定需替换的内容，**

直接输入新内容）

14. 在 Word 2000 中，若要计算表格中某行数值的总和，可使用的统计函数是**（SUM（））**
15. Word 2000 的字数统计功能在**（工具）**菜单中
16. 在 Word 2000 中，需将每页的页码放在页底部右端，正确的操作命令是**（“插入”菜单中的“页码”）**
17. 用快捷方式选定文档中某一段文本的技巧方法是**（把鼠标指针放在文本左端出现选定光标后，三击）**
18. 要给整个页面加一个花纹效果的边框，应该单击“格式”菜单中的“边框与底纹”命令，然后**（单击“页面边框”选项卡，在“线性”栏选择“艺术型”项）**
19. 绘制一个标准的正方形的正确的操作方法是，先在“绘图”工具栏中选择“矩形”，然后**（按住 Shift键用鼠标拖动出正方形）**
20. 把 n 变为 m 的 n 次方，正确的操作是**（先选定 n后，然后设置字体格式为“上标”）**
21. 要在一张表格上套用已有的表格格式，选定表格后正确的操作命令是**（从“表格”菜单中选择“表格自动套用格式”命令）**
22. 给文档分页时，最方便的操作方法是**（按 Ctrl+Enter 键）**
23. 调出“艺术字库”的正确操作是**（单击“绘图”工具栏上的“插入艺术字”图标）**
24. 要想为当前文档中的文字设定字符间距，使用**（“格式”—“字体”菜单项）**
25. 在 Word 2000 中“格式刷”的作用是**（复制格式）**
26. 在 Word 2000 的编辑状态下，对于选定的文字**（即可以移动，也可以复制）**
27. 在 Word 2000 中，新建一个 Word 文档，**Word 会尝试使用文档的第一行文字作为文件名**，但如果第一行文字包含非法字符，或者文件名过长，Word 可能会使用其他方式命名。（此外，某些版本的 Word 可能会直接使用 "文档 1.doc" 作为文件名，而不是自动提取第一行文字。）
28. 对新文档进行保存时，用“文件”菜单中的“保存”与“另存为”命令作用是**相同**的，给旧文档做一个备份时，可用“文件”菜单中的“另存为”命令，“另存为”命令**可将文档保存到其他位置**。

29. 通过“段落”命令调整**“行间距”**还可以调整**“段落前后间距”**，通过字体命令调整**“字符间距”**
30. 调整一页纸所容纳的多少，通常采用的方法有选择**（纸型，页边距）**命令
31. 有关“首字下沉”命令正确的说法**（可以下沉三行字的位置，可悬挂下沉，可以取消“首字下沉”）**
32. 在 Word 2000 文档中选定一个段落的方法是**（在该段落选择区对应出双击鼠标左键，在该段落中三击鼠标左键，按住鼠标左键自段落起始位置拖动到终止位置）**
33. 有关删除表格正确的说法是**（可以删除表格中的某列，可以删除表格中的某行，可以删除单元格）**
34. 选定整个文档的方法有**（按“Ctrl+A”键，在选择区三击鼠标左键，选择“编辑”菜单中的“全选”命令）**
35. 在 Word 2000 中编辑的文档可以保存的格式为**（Word 文档，Html 文档，纯文本）**
36. 给 Word 文档分页的操作方法有**（按“Ctrl+Enter”键，自动分页，选择“插入”菜单中的“分隔符”命令）**
37. Word2000 文档中可插入的对象有**（剪贴画，文本框，艺术字，图表，表格）**

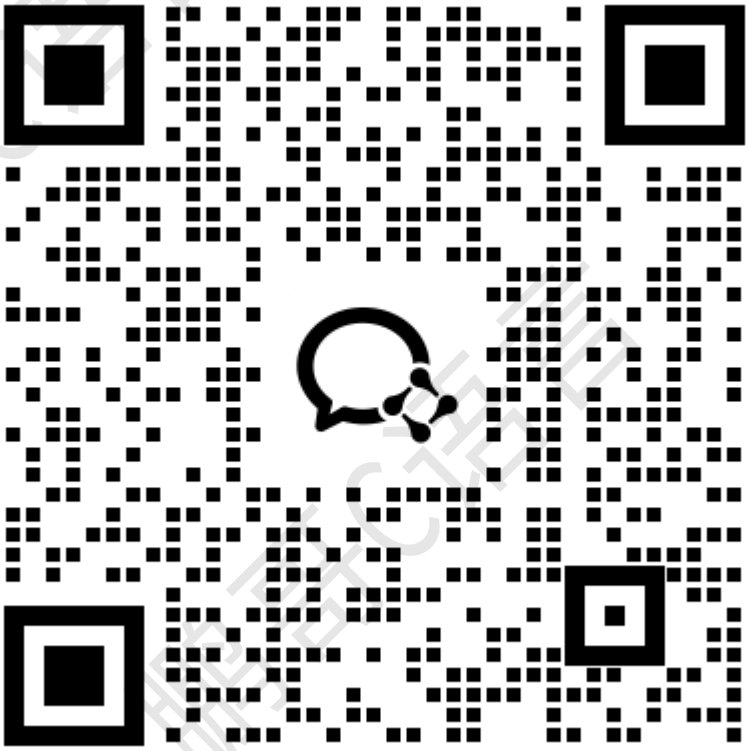


十一、Excel2000 的使用

（共22个高频要点+速记）

1. Excel2000 所属的软件类型为**（电子数据表格软件）**
2. 在 Excel2000 中的工作簿是**（一个 Excel 文档，默认包含 3 个工作表）**- （一个工作簿可以包含任意个工作表）
3. 在 Excel2000 中新建一个电子表格并单击“保存”图标后，默认的文件全名是**（BOOK1.XLS）**
4. 在 Excel 工作簿中同时添加多张工作表操作时应先按住**（Shift）**，活动工作表**（只能有一个）**
5. Excel 的工作表的活动单元格的位置可以**（由所在行和列共同定义）**
6. 在单元格中输入数值超长时**（以科学记数法显示）**
7. 向 Excel 的工作表单元格中输入数据，可以直接将光标定位在编辑栏或对当前活动单元格按**（F2）**，单元格中可以包含**（数字，文本，日期，公式等）**，输入内容后，如果将光标右移一列则按**（Tab）**键
8. 在 Excel 中向单元格输入“4/7”， Excel2000 会将其视作**（日期“4 月 7 日”）**若想输入数值“4/7”应输入**（’ 4/7）**
9. 在 Excel2000 中，表示输入的是一个计算公式，则在输入公式前必须先输入**（等于号=）**
10. Excel 中要在“Sheet2”工作表中引用“Sheet1”工作表 A2 单元格的数据，应使用公式**（=Sheet！A2）**
11. 在 Excel 中，要返回一组参数的最大值，应该使用的函数为**（MAX）**
12. 在 Excel 中，工作表 A2 单元格中有绝对引用“=AVERAGE(C3C$6)”，把它复制到 A3 单元格，则该单元格中的公式是**（和原来不变“=AVERAGE(C3C$6)”）**这个是范围平均值的公式， 意思是从 C3 到 C6的和的平均值，冒号是区域左上角单元格引用和 区域右下角单元格引用之间的符号
13. 在 Excel 中进行重排窗口，其窗口排列方式包括**（水平并排，平铺，垂直并排，层叠）**
14. 在 Excel 单元格中输入等号和公式后，确认输入应**（按 Enter 键，单击编辑栏左侧的对勾）**
15. 在 Excel2000 公式中可以包含的运算符合元素为**（+，-，*，/，及数字；单元格引用；函数；冒号；逗号）**
16. 在 Excel 中，修改公式可以使用的方法有**（选定单元格，单击“=”按钮；选定单元格，在编辑栏直接修改；先双击有关公式的单元格，再进行修改）。**其中（MAX 为最大值， MIN 为最小值，AVERAGE 为平均值，COUNT 为合计，SUM 为总和）
17. 在 Excel 的“工具”—“保护”菜单项中，使用“保护工作表”命令，可以保护的工作表中的元素是**（内容，方案，对象）**
18. 在 Excel 中如果按汉字排序，可以依据的方法为**（笔画多少，按汉语拼音的字母顺序）**

19. 在 Excel 中可对数据清单进行的操作为**（排序，筛选，查询，分类汇总，添加记录）**
20. 下面对 Excel 工作表筛选操作描述正确的有**（可以将特定的数据筛选出来，可以将不符合条件的数据隐藏起来，可以自定义筛选的条件，可以定义多重条件）**
21. 在对 Excel 工作表进行汇总操作之前，应首先进行的操作为**（定位单元格，排序）**
22. 在对 Excel 工作表删除汇总结果时，可以采用的操作为**（选择“数据”—“分类汇总”—“全部删除”；将汇总的数据行删除）**



十二、PowerPoint 的使用

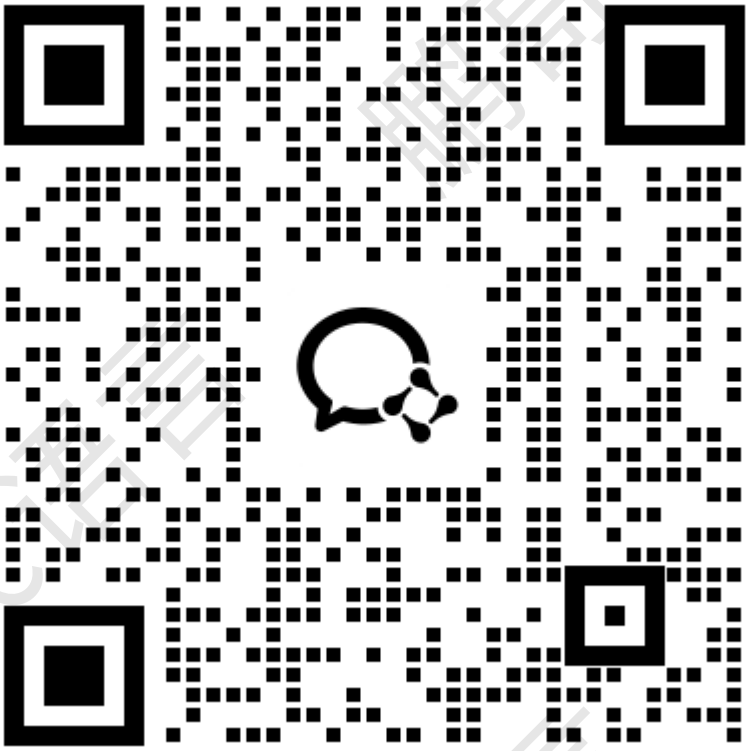
（共41个高频考点+考前速记）

1. PowerPoint2000 所生成的演示文稿的默认扩展名是**（“.ppt”）**，在 PowerPoint2000 中，演示文 稿和幻灯片之间的关系是**（一个演示文稿由多张幻灯片组成）**演示文 稿最基本的组成单位是**（幻灯片）**。
2. 对幻灯片的背景进行设置，应使用的菜单是**（格式）**，（插入项目符号和编号不能从“插入”菜单实现）

3. 在 PowerPoint2000 中可以同时显示多张幻灯片，且能方便地对选中的幻灯片进行移动，复制，删除等操作的视图是**（幻灯片浏览视图）**， 要想在幻灯片中加入演讲者的注释，需要使用的视图是**（备注页视图）**。
4. 在 PowerPoint2000 中放映幻灯片的快捷键是**（F5）**，结束幻灯片的快捷键是**（Esc）**
5. 在 PowerPoint2000 中，改变当前幻灯片的布局，使用的操作是**（选择“格式”——“幻灯片版式”）**，PowerPoint2000 为每张幻灯片提供了各种版式，这些版式中的幻灯片布局主要是通过（占位符）实现的。
6. 在 PowerPoint2000 中建立超链接可以通过**（“插入”—“超链接”；“幻灯片放映—动作设置”；“幻灯片放映—动作按钮”）**

8. 在 PowerPoint2000 现有的幻灯片中，加入已经存盘的演示文告中的幻灯片，应执行的操作是**（选择“插入”—“幻灯片（从文件）” 命令）**
9. 在 PowerPoint2000 中，使每张幻灯片具有统一特征的是通过**（母版）**实现的。属于 PowerPoint2000 母版的是**（标题母版，讲义母版，备注母版）**幻灯片母版（控制着除标题幻灯片以外的所有幻灯片的统一格式；）（选择“视图”菜单中的“母版”命令可以打开幻灯片母版）（创建演示文稿的方式可以通过模板） 注意：幻灯片母版不可以控制是否显示幻灯片编号，“视图”菜单里 的“页眉页脚”命令，才可以（规定幻灯片的编号是否显示）
10. 隐藏幻灯片和删除幻灯片不是同一概念 。
11. 模版和版式不是同一概念
12. 模板和母版不是同一概念
13. PowerPoint2000 配色方案应该使用“工具”菜单
14. 在插入新幻灯片时，新幻灯片的位置在**（当前幻灯片的后一张）**
15. 在幻灯片浏览视图方式下移动幻灯片的方法是**（用鼠标直接拖动）**
16. 幻灯片的版式共有**（28 种）**
17. 插入新幻灯片应该使用命令**（“插入”—“新幻灯片”）**“插入”菜单不能直接插入的是**（文字）**
18. 使用 PowerPoint2000 的超链接，可以使当前位置跳转到**（一个URL，一个电子邮件地址，本地磁盘的一个文档）**建立“超链接”的文字的颜色，可以选择**（“格式”—“幻灯片配色方案”）**
19. 所谓“自定义放映”是指**（自行建立部分幻灯片的一个放映组合）**可以通过自定义放映有选择地放映演示文稿的任意幻灯片 。
20. 通过“幻灯片放映”菜单可以**（设置幻灯片使用的模板）**使用“幻 灯片放映”菜单中的“幻灯片切换”命令，可以**（设置幻灯片的切换 速度；设置幻灯片的切换效果；设置幻灯片切换时的声音效果）**如果 需要为幻灯片加入解说，应使用“幻灯片放映”菜单中的**（录制旁白）**
21. 若想同时选中多张幻灯片，需要使用的视图是**（幻灯片浏览视图）**

22. 每张幻灯片可以有不同的版式，可以有不同的背景，可以有不同的配色方案；（但是不可以有不同的模板）
23. 利用 PowerPoint2000 的“格式”菜单，可以设置幻灯片的版式，可以设置幻灯片使用的模板，可以设置幻灯片的背景**（但是不可以设置幻灯片的切换方式）**
24. 使用“自定义动画”对话框中的**（图标效果）**可以设置幻灯片中图表的动画效果
25. 在 PowerPoint2000“插入”菜单的“图片”子菜单中，包括**（剪贴画，艺术字，组织结构图）**不包括**（图表）**
26. 利用“幻灯片放映”菜单中“动作按钮”**（可以建立超级链接，可以在上面输入文字或出现图片）**
27. 在 PowerPoint2000 的“页面设置”，**（可以设置幻灯片的大小，可以设置幻灯片编号的起始值，可以设置幻灯片的打印方向）**
28. 在 PowerPoint2000 中不可以打印出来的内容是**（母版）**
29. PowerPoint2000 通过（Microsoft Graph）程序，可以在幻灯片中嵌入图表
30. 在 PowerPoint2000 工作区的左下角有五个视图按钮，包含**（幻灯片视图，普通视图，大纲视图，幻灯片浏览视图，幻灯片放映）**没有备注页视图
31. 建立一个“自定义放映”文稿，应该使用的菜单是**（幻灯片放映）**
32. 在 PowerPoint2000“文件”菜单的底部有最近打开过的若干文件名，在默认情况下，通常列出的文件名个数**（4）**
33. 不存在讲义视图，但可以打印幻灯片讲义
34. 同一演示文稿放映时，每张幻灯片的切换方式可以不一致，每张幻灯片配色方案可以不相同，每张幻灯片的放映时间可以不一致；每张幻灯片的背景不一定相同**（但是每张幻灯片模板必须一致）**
35. 可用于在当前演示文稿插入一张新幻灯片的方法是**（使用“插入”菜单中的“新幻灯片命令”，使用“常用”工具栏中的“新建”按钮，使用快接见“Ctrl+M”）**
36. 创建新演示文稿可通过**（内容提示向导，设计模板，空演示文稿）**完成
37. 要设置幻灯片的切换效果，可以通过**（“幻灯片放映”—“幻灯片切换”命令，“幻灯片浏览”工具栏中的“幻灯片切换”按扭）**
38. 属于母板范畴的幻灯片是**（幻灯片母版，讲义母板，标题母板）**
39. 属于全屏幕放映的是**（演讲者放映，在展台浏览）**
40. 选择“视图”—“页眉和页脚”命令，可以设置的显示项目是**（日期和时间，幻灯片编号，页脚）**
41. 使用 PowerPoint2000 的打印命令，可以打印**（全部幻灯片，当前幻灯片，所选定的幻灯片，自定义放映幻灯片，指定编号的幻灯片）**



💡 需要更多C语言资料：如编译器、安装教程、C语言配套视频及刷题资料，扫描下方二维码添加助教领取