1.堆区和栈区的区别：

栈区：存放函数的参数值、局部变量等，由编译器自动分配和释放，通常在函数执行完后就释放了，其操作方式类似于数据结构中的栈。栈内存分配运算内置于CPU的指令集，效率很高，但是分配的内存量有限，比如iOS中栈区的大小是2M

栈区是向低地址扩展的，是一块连续的内存的区域，也就是说，栈顶的地址和栈的最大容量是系统先规定好的，大小在进程分配时是确定的，具体大小看编译器，操作系统。

堆区：就是通过new、malloc 、realloc 、分配的内存块，编译器不会负责它们的释放工作，需要用程序区释放。分配方式类似于数据结构中的链表，“内存泄漏”通常说的就是堆区。

堆区是向高地址扩展的，是不连续的内存区域（这是由于系统是用链表来存储的空闲内存地址的，自然是不连续的动态分配的），因为会手动进行分配，会大一些，大小不固定

2.简述野指针的可能原因和相应的避免方法

野指针是指一个指针变量指向未知或无效的内存地址，或者指向已经被释放的内存区域。使用野指针会导致程序崩溃或产生不可预测的行为，因为野指针指向的内存可能已经被其他程序或操作系统重用，从而导致数据被破坏或泄露

野指针通常的产生原因：

1. 指针变量没有初始化。如果一个指针变量没有被初始化，那么它的值就是未知的，可能指向任何内存地址
2. 释放已经被释放的内存。如果在程序中释放了一小块内存，但没有将该内存的指针设为NULL，那么这个指针就成为了野指针。

要避免野指针，可以采取以下措施：

1. 初始化指针。在声明指针变量时，初始化为NULL或者其他有效值，确保它不是一个野指针
2. 及时释放指针。在使用完指针变量指向的内存后，及时释放内存，并将指针设为NULL
3. 避免指向栈上的变量。如果必须指向栈上的变量，确保指针的作用域在变量作用域内，不要再变量作用域结束后继续使用指针
4. 使用动态内存分配。使用malloc或new分配内存时，确保及时释放内存，并将指针设为NULL
5. 尽量使用智能指针。智能指针可以在指针指向的内存不再需要时自动释放内存，并将指针设为NULL

3.分析程序运行后的输出结果：

&arr类型是指向5个整形数据组成的一维数组的指针，&arr+0x1实际上加了20 后来有转换回int \* 所以ptr1[-1]=5

若arr 假设为0x1000 则0x1000到0x1004 存储0x00000001 0x1004到0x1008存储0x00000002 所以int(arr)+0x1为0x1001 \*ptr取的是0x1001到0x1005中存储的数据 也就是（小端序）00 00 00 02 故输出为0x02000000

4.枚举的优点：和#define定义的标识符相比枚举有类型检查，更加严谨；便于调试，#define定义的常量不便于调试；使用方便，可以一次定义多个常量