1. 堆和栈的区别：
2. 堆由程序员手动管理，栈是由编译器自动管理
3. 在堆中，频繁new/delete会造成内存空间不连续，产生内存碎片；栈是先进后出的结构，不存在内存碎片；
4. 对于堆来说，生长方向是内存地址增加的方向，而栈的生长是内存地址减小的方向
5. 堆都是动态分配的，不存在静态分配的堆；栈可以动态分配和静态分配，静态分配包括局部变量，编译器进行分配，动态分配通过alloc函数进行分配
6. new和delete的实现；
7. new：针对简单数据类型直接调用operate new分配内存；失败直接抛出异常；针对复杂类型，先调用operate new，再在分配的内存上调用构造函数；
8. delete：针对简单数据类型，直接调用free；针对复杂数据类型先调用析构函数，再调用operate delete；
9. 与mallco/free的区别：

（1）属性上：new / delete 是c++关键字，需要编译器支持。 malloc/free是库函数，需要c的头文件支持。  
（2）参数：使用new操作符申请内存分配时无须制定内存块的大小，编译器会根据类型信息自行计算。而mallco则需要显式地指出所需内存的尺寸。  
（3）返回类型：new操作符内存分配成功时，返回的是对象类型的指针，类型严格与对象匹配，故new是符合类型安全性的操作符。而malloc内存成功分配返回的是void \*，需要通过类型转换将其转换为我们需要的类型。  
（4）分配失败时：new内存分配失败时抛出bad\_alloc异常；malloc分配内存失败时返回 NULL。  
（5）自定义类型：new会先调用operator new函数，申请足够的内存（通常底层使用malloc实现）。然后调用类型的构造函数，初始化成员变量，最后返回自定义类型指针。delete先调用析构函数，然后调用operator delete函数释放内存（通常底层使用free实现）。 malloc/free是库函数，只能动态的申请和释放内存，无法强制要求其做自定义类型对象构造和析构工作。  
（6）重载：C++允许重载 new/delete 操作符。而malloc为库函数不允许重载。  
（7）内存区域：new操作符从自由存储区（free store）上为对象动态分配内存空间，而malloc函数从堆上动态分配内存。其中自由存储区为：C++基于new操作符的一个抽象概念，凡是通过new操作符进行内存申请，该内存即为自由存储区。而堆是操作系统中的术语，是操作系统所维护的一块特殊内存，用于程序的内存动态分配，C语言使用malloc从堆上分配内存，使用free释放已分配的对应内存。自由存储区不等于堆，如上所述，布局new就可以不位**于堆中。**