C++ 贪心算法（上） 课后习题

姓名: _____

一、 填空题

1. 贪心算法在求解问题的每一步中，都做出一个 _____ 的选择，寄希望于一系列的局部最优选择能够产生一个 _____。
2. 一个问题能用贪心算法求解，通常需要满足 _____ 和最优子结构两个关键性质。
3. 在部分背包问题中，正确的贪心标准是优先选择 _____ 最高的物品。
4. 在C++代码实现中，为了对物品按价值密度进行排序，我们可以使用`sort`函数，并为其提供一个自定义的 _____。
5. 一个问题的最优解包含了其子问题的最优解，这个性质被称为 _____。

二、 单项选择题

( ) 1. 贪心算法的核心思想是?
A. 穷举所有可能 B. 每步都做出局部最优选择 C. 将问题分解为不相关的子问题 D. 从后向前推导最终解

( ) 2. 根据讲义，以下哪个问题是适合用贪心算法解决的典型例子?
A. 0-1背包问题 B. 排序问题 C. 部分背包问题 D. 二分问题

( ) 3. 对于部分背包问题，最佳的贪心策略是?
A. 优先选择价值最高的物品 B. 优先选择重量最轻的物品 C. 优先选择价值密度最高的物品 D. 随机选择物品

( ) 4. “某个状态以前的过程不会影响以后的状态，只与当前状态有关”描述的是贪心算法的哪个性质?
A. 最优子结构 B. 无后效性 C. 高效性 D. 启发性

( ) 5. 假设背包容量为20，有A、B两个物品。A重10价值60，B重20价值100。按照部分背包问题（可拆分物品）的贪心策略，最终得到的最大价值是?
A. 120 B. 110 C. 100 D. 60

( ) 6. 在部分背包问题的C++代码实现中，`Item`结构体中的`ratio`（价值密度/性价比）成员最好使用什么类型?
A. `int` B. `char` C. `bool` D. `double`

( ) 7. 已知`Item`结构体内有三个成员变量分别为：`v`: 物品价值，`m`: 物品重量，`ratio`: 物品价值密度, 在对物品按价值密度进行降序排序时，自定义比较函数`bool cmp(const Item& a, const Item& b)`的正确写法是?
A. `return a.v > b.v;` B. `return a.m < b.m;` C. `return a.ratio > b.ratio;` D. `return a.ratio < b.ratio;`

( ) 8. 一个贪心策略通常可以使用什么办法来证明?
A. 公式法 B. 暴力枚举法 C. 反证法 D. 数学归纳法

( ) 9. 在处理最后一个无法完全装入的物品时，应该如何计算其贡献的价值?
A. 计为0 B. 计为其全部价值 C. (剩余容量 / 物品重量) * 物品价值 D. 无法计算

( ) 10. 为什么“优先选择价值最高的物品”的策略对于部分背包问题是错误的?
A. 因为它忽略了物品的重量 B. 因为它忽略了背包的容量 C. 因为价值最高的物品总是最重 D. 因为该策略计算过于复杂

三、 判断题 (对的打“√”，错的打“×”)

( ) 1. 贪心算法适用于所有最优化问题，并且总能找到最优解。

( ) 2. 对于部分背包问题，我们可以将物品分割成任意大小的块装入背包。

( ) 3. 在贪心算法中，一旦一个选择被做出，它就不会被撤销或更改。

( ) 4. “重量最轻优先”是解决部分背包问题的一个有效贪心策略。

( ) 5. 贪心算法就算得不到最优解，也能够快速给出一个质量较高的可行解。

( ) 6. 贪心算法的正确性通常需要严格的数学证明，不能仅凭直觉。

( ) 7. 在C++代码中，`sort(items.begin(), items.end(), cmp);` 这行代码实现了贪心策略的核心部分，即对物品进行排序。

( ) 8. 如果一个物品的重量大于当前背包的剩余容量，那么该物品以及之后的所有物品都不能再装入背包。

( ) 9. 贪心算法相较于暴力搜索算法，通常具有更高的时间复杂度。

( ) 10. 在计算价值密度时，`ratio = (double)v / m;` 是为了防止整数除法导致精度丢失。