解法一

该算法与课本中基本的 Gale-Shapley 算法非常相似。在任何时间点，一个学生要么“承诺”给一所医院，要么是“自由的”。一所医院要么有可用职位，要么是“满的”。该算法如下：

Algorithm 1 医院-学生匹配算法

while 医院 h_i 还有可用职位 **do**
  h_i 向下一个学生 s_j （根据其偏好列表）提供职位
  if s_j 是自由的 **then**
    s_j 接受该提议
  else ▷ s_j 已经承诺给医院 h_k
    if s_j 偏好 h_k 超过 h_i **then**
      s_j 保持对 h_k 的承诺
    else
      s_j 变为承诺给 h_i
      医院 h_k 的可用职位数量增加一。
      医院 h_i 的可用职位数量减少一。
    end if
  end if
end while

该算法在 $O(mn)$ 步内终止，因为每所医院都会向一名学生提供一个职位，并且在每次迭代中，一些医院会向一些学生提供职位。假设医院 h_i 有 $p_i > 0$ 个可用职位。当所有医院的职位都被分配完毕时，算法终止，因为任何医院都不会填满所有职位，必须向每个学生提供一份职位；但是，所有这些学生都将承诺给某些医院，这与我们的假设 $\sum_{i=1}^{m} p_i < n$ 相矛盾。

最后，我们想论证该分配是稳定的。对于第一种不稳定情况，假设存在学生 s 和 s'，以及医院 h，并且 h 偏好 s' 超过 s，那么 h 会在向 s 提供职位之前向 s' 提供职位；从那时起，s' 将承诺给某个医院，因此不会是自由的——这是一个矛盾。

对于第二种不稳定情况，假设 (h_i, s_j) 是一对导致不稳定的情况。那么 h_i 一定已经向 s_j 提供了一个职位，否则它会偏好所有它承诺给 s_j 的居民。此外，s_j 一定拒绝了 h_i 而选择了某个 h_k，她/他更喜欢 h_k；并且 s_j 必须承诺给某个 h_l（可能与 h_k 不同），她/他/它也偏好 h_i。

解法二

我们考虑一个经典的稳定匹配问题，其中有 n 名学生 $S = \{s_1, \ldots, s_n\}$ 和 m 所医院 $H = \{h_1, \ldots, h_m\}$。每所医院 h_i 拥有 $c_i > 0$ 个可用职位（容量），且所有医院的总容量为 $\sum_{i=1}^{m} c_i$。每名学生对所有医院（包括可能存在的虚拟医院）有一个偏好列表，每所医院对其可能录取的学生也有一个偏好列表。我们的目标是找到一个稳定的匹配。

为了将这个带容量的匹配问题转化为标准的单对单稳定匹配问题，我们引入一个**虚拟医院** h_0。

算法构造

1. **虚拟医院的定义**：构造一个虚拟医院 h_0。

- **容量**：设定 h_0 的容量为 $C_{h_0} = n - \sum_{i=1}^{m} c_i$。这个容量确保了所有学生都能在最终获得一个“职位”，无论是在真实医院还是在虚拟医院。
- **学生偏好**：在所有学生的偏好列表中，虚拟医院 h_0 排在所有真实医院之后，即对于任意学生 $s_j \in S$，如果 h_k 是任意真实医院，h_0 总是比 h_k 排序更靠后。这意味着学生只有在被所有真实医院拒绝后，才会考虑 h_0。
- **虚拟医院偏好**：虚拟医院 h_0 对所有学生一视同仁（例如，按照某个预设的固定顺序，或者无偏好），只要能填补其容量。

2. **问题转化**：通过引入 h_0，我们将问题转化为一个拥有 n 名学生和 $m+1$ 所医院（包括 h_0）的稳定匹配问题，其中：

   - 真实医院 h_i 的 c_i 个职位可以视为 c_i 个独立的、对学生有相同偏好的“分身”医院。
   - 虚拟医院 h_0 也有其 C_{h_0} 个“分身”。

   现在，每个学生将与一个（且仅一个）医院的“分身”进行匹配，而每个医院的“分身”也只会匹配一个学生。这完美契合了 Gale-Shapley 算法的框架。

3. **应用 Gale-Shapley 算法（学生提出申请版本）**：我们现在可以使用经典的学生提出申请的 Gale-Shapley 算法。

Algorithm 2 **带虚拟医院的 Gale-Shapley 算法**

初始化所有学生未被匹配。
初始化所有医院（包括 h_0）的所有职位为空闲。
while 存在**未被匹配**的学生 s **do**
    s 向其偏好列表中下一个尚未向其提出申请的医院 H' 提出申请。
    if H' 的某个职位**空闲** **then**
        H' **临时接受** s。
    else ▷ H' 的所有职位都已被临时接受
        令 s' 是 H' 当前临时接受的学生中，根据 H' 的偏好列表，H' **最不喜欢**的一个学生。
        if H' **偏好** s 胜过 s' **then**
            H' **拒绝** s'。
            s' 变为**未匹配**状态。
            H' **临时接受** s。
        else
            H' **拒绝** s。
        end if
    end if
end while
返回最终的匹配结果。

算法分析与稳定性证明

终止性

该算法在有限步内终止。**Gale-Shapley 算法的终止性已经被证明**。由于每次迭代中，要么一个学生成功地向一个新的医院提出申请，要么一个学生被其当前临时匹配的医院拒绝而向下一个医

院提出申请，每个学生向每个医院提出申请的次数是有限的，且每个医院接受的临时匹配数量也有限。因此，整个过程将在 $O(N \cdot M')$ 步内终止，其中 N 是学生总数，M' 是扩展后的医院（包括虚拟医院分身）的总职位数，即 $\sum c_i + C_{h_0} = n$。实际上，更精确的复杂度是 $O(mn)$。

稳定性

该算法产生的匹配是稳定的。我们通过**反证法**来证明：假设存在一对**不稳定**的学生 s^* 和医院 h^*。这意味着：

1. s^* **偏好** h^* 胜过她/他当前匹配的医院 h_{s^*}（如果 h_{s^*} 是虚拟医院 h_0，则 s^* 显然偏好任何真实医院胜过 h_0）。
2. h^* **偏好** s^* 胜过她/他当前匹配的某个学生 s_{h^*}。

根据 **Gale-Shapley 算法的性质**：

- 由于 s^* **偏好** h^* 胜过 h_{s^*}，且学生总是按偏好顺序提出申请，所以 s^* 必然曾经向 h^* **提出过申请**。
- 当 s^* 向 h^* **提出申请**时，有两种情况：
  1. h^* **接受了** s^*：这意味着 s^* 暂时或最终被 h^* 匹配。如果 s^* 后来被 h^* 拒绝了，那一定是 h^* 找到了一个它更偏好的学生来替换 s^*。在这种情况下，h^* 不会偏好 s^* 胜过 s_{h^*}，与假设矛盾。
  2. h^* **拒绝了** s^*：这意味着在 s^* 提出申请时，h^* 的所有职位已被临时占据，并且 h^* 偏好当前占据其职位的学生胜过 s^*。自那以后，h^* 对 s^* 的偏好不会改变，且其职位的学生只会变得更优（或保持不变）。因此，h^* 不会偏好 s^* 胜过 s_{h^*}，与假设矛盾。

这两种情况都导出了**矛盾**，因此，不存在这样的**不稳定对**。尤其值得注意的是，引入**虚拟医院** h_0 统一了不稳定因素的讨论：

- 如果一个学生最终匹配到 h_0，这意味着她/他已经被所有真实医院拒绝，所以她/他不会对任何真实医院有“悔恨”（即偏好某个未录取的真实医院）。
- 虚拟医院 h_0 的特殊偏好（对学生一视同仁或按固定顺序），以及它作为学生偏好列表最末位的地位，确保了它不会参与构成不稳定的对。

因此，该算法产生的匹配是**稳定的**。