题目描述：
有一名科学家想要从一台古董电脑中拷贝文件到自己的电脑中加以研究。但此电脑除了有一个 3.5 寸软盘驱动器以外，没有任何手段可以将文件拷贝出来，而且只有一张软盘可以使用。因此这一张软盘是唯一可以用来拷贝文件的载体。
科学家想要尽可能多地将计算机中的信息拷贝到软盘中，做到软盘中文件内容总大小最大。已知该软盘容量为 1474560 字节。文件占用的软盘空间都是按块分配的，每个块大小为 512 个字节。一个块只能被一个文件使用。拷贝到软盘中的文件必须是完整的，且不能采取任何压缩技术。

输入描述：
第 1 行为一个整数 N，表示计算机中的文件数量。1 <= N <= 1000。
接下来的第 2 行到第 N+1 行（共 N 行），每行为一个整数，表示每个文件的大小 Si，单位为字节。0 <= i < N，0 <= Si
输出描述：
科学家最多能拷贝的文件总大小。
补充说明：
为了充分利用软盘空间，将每个文件在软盘上占用的块记录到本子上。即真正占用软盘空间的只有文件内容本身。
示例 1
输入：
3
737270
737272
737288
输出：
1474542
说明：
3 个文件中，每个文件实际占用的大小分别为 737280 字节、737280 字节、737792 字节。只能选取前两个文件，总大小为 1474542 字节。虽然后两个文件总大小更大且未超过 1474560 字节，但因为实际占用的大小超过了 1474560 字节，所以不能选后两个文件。
示例 2
输入：
6
400000
200000
200000
200000
400000
400000
输出：
1400000
说明：
从 6 个文件中，选择 3 个大小为 400000 的文件和 1 个大小为 200000 的文件，得到最大总大小为 1400000。

也可以选择 2 个大小为 400000 的文件和 3 个大小为 200000 的文件，得到的总大小也是 1400000。

```
import math
h= int(input())
ai= [int(input()) for _ in range(h)]
def kaoBei():
    zijie= 1474560 // 512
    dp = [0]*(zijie+1)
    for i in range(h):
        weight = math.ceil(ai[i]/512.0)
        worth = ai[i]
        for j in range(zijie,weight-1,-1):
            dp[j]= max(dp[j],dp[j-weight]+worth)
    return dp[zijie]

print(kaoBei())
```