①最小堆：heapq
```
import heapq

heapq.heapify(list)
#将列表原地转化为小根堆

heapq.heappush(heap,x)
#将元素 x 加入堆

heapq.heappop(heap)
#弹出并返回堆中最小元素

heapq.heappushpop(heap,x)
#先 push 再 pop，效率高

heapq.nlargest(k,list)
#返回前 k 个最大值

heapq.nsmallest(k,list)
#返回前 k 个最小值
```

②双端队列 deque
```
from collections import deque

deque(list)
#创建一个 deque

append(x)
#在右端添加元素 x

appendleft(x)
#在左端添加元素 x

pop()
#删除并返回右端元素

popleft()
#删除并返回左端元素（常用于 BFS）

extend(list)
#在右端添加列表中的所有元素

extentleft(list)
#在左端添加多个元素（注意顺序会反向）
```

```
remove(x)
#删除第一次出现的元素 x

rotate(k)
#队列元素平移 k 位（k 为正为右移）
```

③字典 defaultdict

```
from collections import defaultdict
```

在访问不存在的键时，自动为该键创建一个默认值

```
d=defaultdict(int/list/set/str)
```

④修改递归深度 sys

```
import sys

sys.setrecursionlimit(10000)
#  将最大递归深度设置为  10000
```

⑤记忆化 lru_cache

```
from functools import lru_cache

@lru_cache(maxsize=None)#默认缓存结果无上限

def fib(n):
    if n<2:
        return n
    return fib(n-1)+fib(n-2)

print(fid(100))
#能够极大简化时间复杂度
```