单源最短路（P4779）

```python
from Yawn_Sean_Template import *
def main():
    n,m,s=MII()
    e=lst_lst(n+1)
    for _ in range(m):
        u,v,w=MII()
        e.append(u,(v,w))
    dis=[inf]*(n+1)
    dis[s]=0
    heap=[]
    heappush(heap,(0,s))
    while heap:
        d,u=heappop(heap)
        if d!=dis[u]:
            continue
        for v,w in e.iterate(u):
            nd=d+w
            if dis[v]>nd:
                dis[v]=nd
                heappush(heap,(nd,v))
    for i in range(1,n+1):
        print(dis[i],end=' ')
    return
```

求树的直径（B4016）

`dfs` 前用 `@bootstrap` 修饰， `dfs` 内部递归和返回时用 `yield`

```python
from Yawn_Sean_Template import *
def main():
    n=II()
    e=lst_lst(n+1)
    for _ in range(n-1):
        u,v=MII()
        e.append(u,v)
        e.append(v,u)
    @bootstrap
    def dfs(u,fa) -> Tuple[int,int]:
        nd,l=u,1
        for v in e.iterate(u):
            if v==fa:
                continue
            tnd,tl=yield dfs(v,u)
            tl+=1
            if tl>l:
                nd=tnd
                l=tl
        yield (nd,l)
    st,_=dfs(1,0)
    ed,ans=dfs(st,0)
    print(ans-1)  # 经过了 ans 个节点, ans-1 条边
    return
```

哈希使用:

```
from Yawn_Sean_Template import *
def main():
    dict=defaultdict(int)  # 创建字典
    dict[Wrapper(num)]=num
    key=Wrapper(num)
    if ket in dict:
        print("find key num")
    else:
        print("key num dosen't exist")
```

排序（以从大到小排序为例）：

```python
from functools import cmp_to_key
"""
-1: x 排在 y 前
0:  x 和 y 相同
1:  x 排在 y 后
"""
def cmp(x,y):
    if x==y: return 0
    return -1 if x>y else 0

a=LII()
a.sort(key=cmp_to_key(cmp))
b=sort(a,key=cmp_to_key(cmp))
```

这个排序跟 C++ 不同，C++ 是:

- `cmp(x,y)==1`，则 x 严格排在 y 前
- `cmp(x,y)==0`，则 x 和 y 相同或者严格排在 y 后