习题二、2.试写出 0/1 背包问题的队列式分支限界算法程序，并找一个物品个数是 16 的例子检验程序的运行情况。
代码:

```
#include<stdio.h>
#include<iostream>
#include<queue>          //优先队列头文件
using namespace std;
#define maxNum 20        //物品最大值
typedef struct Node
{
    int ub;
    int w;
    int v;
    int index;
    int object[maxNum]={0};
    bool operator < (const Node &p)const     //大根堆
    {
        p.ub>ub;
    }
}Node, *QNode;
int w[maxNum];           //存放物品
int v[maxNum];           //存放价值
int w_v[maxNum];         //存放性价比
int list[maxNum] = {0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19};
//物品排序映射
priority_queue<Node> pp;
int W;                   //背包容量
int up;                  //上界
int down;                //下界
int n;                   //物品数
void sort(int i,int j)     //对存储数组由大到小排序，此处采用快速排序
{
    int start=i;
    int end=j;
    if(start>=end)
    {
        return;
    }
    while(start!=end)
    {
        int temp;
        int temp1;
        int temp2;
        int t;
```

```
        while(start<end&&w_v[start]>w_v[end])
        {
        end--;
        }
        if(start==end)
        break;
        temp=w_v[start];
        temp1=w[start];
        temp2=v[start];
        w_v[start]=w_v[end];
        w[start]=w[end];
        v[start]=v[end];
        w_v[end]=temp;
        w[end]=temp1;
        v[end]=temp2;
        t = list[start];
        list[start] = list[end];
        list[end] = t;
        start++;
        if(start==end)
        break;
        while(start<end&&w_v[start]>w_v[end])
        {
            start++;
        }
        if(start==end)
        break;
        temp=w_v[start];
        temp1=w[start];
        temp2=v[start];
        w_v[start]=w_v[end];
        w[start]=w[end];
        v[start]=v[end];
        w_v[end]=temp;
        w[end]=temp1;
        v[end]=temp2;
        t = list[start];
        list[start] = list[end];
        list[end] = t;
        end--;
    }
    sort(i,start-1);
    sort(end+1,j);
    return;
```

```
}
void get_down()        //获取下界
{
    down=0;
    int wight=0;
    for(int i=0;i<n;i++)
    {
        wight=wight+w[i];
        if(wight<=W)
        {
            down=down+v[i];
        }
        else
        break;
    }
    return;
}
void get_up()           //获取上界
{
    up=w_v[0]*n;
    return;
}
int get_ub(Node node)        //获取目标函数值
{
    node.ub=node.v+w_v[node.index+1]*(W-node.w);
    return node.ub;
}
Node solve()
{
    Node result;
    get_down();
    get_up();
    Node first;      //定义起点
    first.index=-1;
    first.ub=up;
    first.v=0;
    first.w=0;
    pp.push(first);
    int ret=0;
    while(pp.size())
    {

        Node tmp=pp.top();
        pp.pop();
```

```
        if(tmp.index==n-2)
        {
            int ans=tmp.v;
            if(tmp.w+w[tmp.index+1]<=W)
            {
                ans=ans+v[tmp.index+1];
                tmp.object[tmp.index+1] = 1;
            }
            if(ans>=tmp.ub)      //价值大于全部目标函数值，直接得出最优解
            {
                ret=ans;
                result=tmp;
                break;
            }
            else
            {
                if(ans>down)          //更新下界
                down=ans;
                if(ans>ret)
                ret=ans;
                result=tmp;
                continue;
            }
        }
        Node next;
        for(int i=0;i<2;i++)
        {
            next.index=tmp.index+1;
            next.v=tmp.v+i*v[tmp.index+1];
            next.w=tmp.w+i*w[tmp.index+1];
            next.ub=get_ub(next);
            for(int j=0;j<tmp.index+1;j++)
                next.object[j] = tmp.object[j];
            next.object[next.index] = i;
            if(next.w<=W&&next.ub>=down)      //大于下界且容量小于背包容量入
队列
                    pp.push(next);
        }
    }
  printf("最优解为: %d\n",ret);
  return result;
}
main()
{
```

```
    Node result;
    printf("请输入背包容量: ");
    scanf("%d",&W);
    printf("请输入物品数量: ");
    scanf("%d",&n);
    for(int i=0;i<n;i++)
    {
        printf("请输入物品重量和价值（格式如 1 2 ）: ");
        scanf("%d %d",&w[i],&v[i]);
        w_v[i]=v[i]/w[i];
        if(v[i]%w[i]!=0)
        w_v[i]+=1;
    }
    sort(0,n-1);
    result = solve();
    printf("\n 选择的物品: \n");
    for(int j=0;j<n;j++)
    {
        if (result.object[j])
            printf("物品 %d\n",list[j]+1);
    }
    //list:物品被排序后，如果0=list[2]，那么表示物品 0 被映射到 2 处。
i=list[j]，表示物品 i 被映射到 j 处
    //object: 是否选择 j 位置的物品
    //如果 object[j]=1，意味着原位置 i=list[j]的物品被选择，即物品 i。
}
```

16 个物品的测试:

```
ubuntu@VM-8-5-ubuntu:~/lhc/test$ ./test
请输入背包容量: 16
请输入物品数量: 16
请输入物品重量和价值（格式如 1 2 ）: 1 2
请输入物品重量和价值（格式如 1 2 ）: 1 3
请输入物品重量和价值（格式如 1 2 ）: 1 4
请输入物品重量和价值（格式如 1 2 ）: 2 2
请输入物品重量和价值（格式如 1 2 ）: 2 3
请输入物品重量和价值（格式如 1 2 ）: 2 4
请输入物品重量和价值（格式如 1 2 ）: 3 5
请输入物品重量和价值（格式如 1 2 ）: 3 4
请输入物品重量和价值（格式如 1 2 ）: 3 6
请输入物品重量和价值（格式如 1 2 ）: 4 1
请输入物品重量和价值（格式如 1 2 ）: 4 2
请输入物品重量和价值（格式如 1 2 ）: 4 3
请输入物品重量和价值（格式如 1 2 ）: 4 4
请输入物品重量和价值（格式如 1 2 ）: 4 5
请输入物品重量和价值（格式如 1 2 ）: 4 6
请输入物品重量和价值（格式如 1 2 ）: 5 2
最优解为: 29

选择的物品:
物品 3
物品 2
物品 9
物品 15
物品 14
物品 1
物品 5
```