①算法实现题4-6 最优服务次序问题

问题描述：设有n个顾客同时等待―项服务。顾客i需要的服务时间为ti，1≤i ≤n。应如何安排n个顾客的服务次序才能使平均等待时间达到最小？平均等待时 间是n个顾客等待服务时间的总和除以n。

编程任务：对于给定的n个顾客需要的服务时间，编程计算最优服务次序。

解：代码如下：

#include<bits/stdc++.h>

using namespace std;

int main()

{

int n,sum=0;

cin >> n;

double ave;

int a[n];

for(int i=0;i<n;i++)

{

cin >> a[i];

}

sort(a,a+n);

for(int i=0;i<n;i++)

{

sum=sum+a[i]\*(n-i);

}

/\*

for(int i=1;i<n;i++)

{

a[i]=a[i]+a[i-1]; // 第i+1个顾客等待时间应该是第i个顾客等待时间+第i+1个顾客花费的时间a[i+1]

}

for(int i=0;i<n;i++)

{

sum=sum+a[i];

}

\*/

ave=sum\*1.0/n;

printf("%.2f",ave);

return 0;

}

②算法实现题4-7 多处最优服务次序问题

问题描述：设有n个顾客同时等待一项服务。颜客i需要的服务时间为ti，l≤i ≤n。共有s处可以提供此项服务。应如何安排n个顾客的服务次序才能使平均 等待时间达到最小？平均等待时间是n个顾客等待服务时间的总和除以n。

编程任务：对于给定的n个顾客需要的服务时间和s的值，编程计算最优服务次序。

解：代码如下：

double greedy(vector<int> client, int s)

{

//服务窗口的顾客等待时间

vector<int> service(s+1, 0);

//服务窗口顾客等待时间的总和

vector<int> sum(s+1, 0);

//顾客的数量

int n = client.size();

//按顾客的服务时间升序排序

sort(client.begin(), client.end());

//贪心算法的实现

int i=0; //顾客的指针

int j=0; //窗口的指针

while(i < n)

{

service[j] += client[i];

sum[j] += service[j];

++i, ++j;

if(j == s) j = 0;

}

//计算所有窗口服务时间的总和

double t=0;

for(i=0; i<s; ++i) t += sum[i];

t /= n;

return t;

}

#include <iostream>

#include <vector>

using namespace std;

double greedy(vector<int> client,int s)

{

vector<int> service(s+1,0);

vector<int> sum(s+1,0);

int n = client.size();

sort(client.begin(),client.end());

int i=0, j=0;

while(i < n)

{

service[j] += client[i];

sum[j] += service[j];

++i,++j;

if(j == s) j = 0;

}

double t=0;

for(i=0; i<s; ++i) t += sum[i];

t /= n;

return t;

}

int main()

{

vector<int> queue;

int n, s;

scanf("%d%d", &n, &s);

int temp;

for(int i=0; i<n; i++)

{

scanf("%d", &temp);

queue.push\_back(temp);

}

printf("%.3lf\n", greedy(queue, s));

}