问题背景：

全球疫情形势严峻，各国医用物资、生活物资的调配需求强烈。小明所在的公司有一批不同类型的卡车，他想设计程序，根据不同类型的卡车安排不同的运送任务，并总体计算收益。

任务描述：

设计不同的类来描述不同类型的卡车，并为公司设计一个管理类，对卡车进行管理和分配。

卡车类型有如下三类，每类卡车有不同的成本计算和费用计算方式（记运送公里数为d，运送货物吨数为w）：

1. 普通卡车（NormalTruck）运送成本低，所以起步价低、里程价低，但是**有运输里程限制**，超出限制的任务不能进行运送。

- 成本计算：w * d
- 费用计算：按照每公里5元计费，但当费用小于100元时，则收取最低100元。

2. 高级卡车（AdvancedTruck）运送成本高，所以起步价高、里程价高，但是可以进行比较长距离的运送（没有运输里程限制）。

- 成本计算：50 + w * d
- 费用计算：按照每公里8元计费，但当费用小于150元时，则收取最低150元。

3. 长距离卡车 （LongDistanceTruck）一部分高级卡车更希望进行长距离运送，因此它们的计费方式在高级卡车的基础上进行了一些调整。

- 成本计算：与高级卡车相同
- 费用计算：按照高级卡车的最终计费，如果运程小于30公里，价格计为原先的1.1倍，运程等于或超过30公里时，价格计为原先的9折。

公司管理一批不同类型的卡车，根据每一个接到的运送需求（包含运送公里数和货物吨数）安排可以运送的计费最低的卡车进行运送。

设计要求

卡车类

为便于使用，定义Truck类为所有卡车类的基类，提供计算成本（cost）、计算费用（price）、执行任务（transport）、累计成本（getTotalCost）、累计收入（getTotalIncome）四个函数：

```
Truck(int id); //每辆卡车有一个唯一的id
double cost(int targetDistance, int weight); //计算成本，超过运输里程限制则返回-1
double price(int targetDistance); //计算费用，超过运输里程限制则返回-1
void transport(int targetDistance, int weight);//执行给定的任务
double getTotalCost() const; //该辆卡车完成运输任务所花费的累计成本
double getTotalIncome() const; //该辆卡车完成运输任务所获得的累计收入
int getID()const; //返回该车的ID;
```

为三类卡车设计并实现对应的类（作为Truck的派生类，或者派生类的派生类），并提供各自的构造函数和Truck类中所规定的功能。

```
NormalTruck(int id, int maxDistance);
AdvancedTruck(int id);
LongDistanceTruck(int id);
```

管理类

管理类管理一系列的卡车，并提供以下函数：

```
Manager:: Manager();
void Manager:: addTruck(Truck* t);//传入对象的生命周期在程序功能完成前不会结束
Truck* Manager:: transport(int targetDistance,int weight);//根据指定的运送公里数和运送货物吨数寻找合适的价格（price）最低的卡车，并执行运送任务,返回执行任务卡车的指针。测试时保证每个任务至少存在有一个卡车可以执行运输任务。
//如果价格相同，选取累计收入最小的卡车；如果累计收入也相同，则选取id编号最小的卡车。
//执行运送任务应调用对应对象的transport方法。
double Manager::getAllIncome();//返回累计总收入
double Manager::getAllCost();//返回累计总成本
```

测试调用接口：

所有在设计要求中定义的类的成员函数

数据说明

Manager管理的Truck数量最大不超过40;

cost和price小于100000;

测试时数据保证所有任务都至少有一辆卡车可以执行;

测试用例保证所有的运算都不会溢出。

注意事项

1.你需要完成对应要求，并创建以下**两个文件**，将其打包为zip压缩包上传；

```
xxxx.zip
  |--Transport.h  //所有卡车类的定义都放在transport.h中
  |--Transport.cpp  //所有卡车类的成员函数的实现放在transport.cpp中
  |--Manager.h //Manager类定义放在manager.h中
  |--Manager.cpp //Manager类成员函数的实现放在manager.cpp中
```

2.你可以在类中增加新的函数，成员变量，但不要改变类的命名及接口，否则无法通过测试；

3.注意内存安全，避免内存泄漏；

4.注意文件编码格式为utf-8；

5.注意不要在提交的源代码中包含main函数；