Весёлый паровозик

Напишите класс Train, описывающий движение паровоза по рельсам. Паровоз описывается тремя параметрами - масса, текущая скорость, текущая координата (одномерная, отсчитывается вдоль рельсов).

Прототип класса:

```
class Train
{
public:
    // Создать паровоз массой m,
    // стоящий в начале координат
    Train(double m);

    // Ехать с текущей скоростью в течение времени dt
    void move(double dt);

    // Изменить полный импульс паровоза (p = mv) на dp
    // Изменение может менять знак скорости
    void accelerate(double dp);

    // Получить текущую координату паровоза
    double getX();
};
```

Код для базового тестирования реализации класса:

```
    Train t(10);
    t.accelerate(1); // Скорость стала 0.1
    t.move(1);
    cout << "Current X: " << t.getX() << endl;
    t.move(1);
    cout << "Current X: " << t.getX() << endl;
    t.accelerate(-2); // Скорость стала -0.1
    t.move(3);
    cout << "Current X: " << t.getX() << endl;
```

Базовый тест должен вывести:

```
Current X: 0.1
Current X: 0.2
Current X: -0.1
```

Газгольдер

Напишите класс GasHolder, описывающий состояние термостатированного сосуда с газом. Считать, что для газа верно уравнение состояния PV = (m/M)RT. Значение постоянной R принять 8.31 Дж/(моль*К).

Прототип класса:

```
class GasHolder
{
public:
    // Создать газгольдер заданного объёма.
    // Температура созданного термостата равна 273 К.
    GasHolder(float v);

    // Уничтожить газгольдер.
    ~GasHolder();

    // Впрыск порции газа массой m и молярной массой M.
    // Считать, что газ принимает текущую температуру газгольдера за
пренебрежимо малое время.
    void inject(float m, float M);

    // Подогреть газгольдер на dT градусов.
    // Считать, что нагрев возможен до любых значений температуры.
    void heat(float dT);

    // Охладить газгольдер на dT градусов.
    // При попытке охладить ниже 0 К температура становится ровно 0 К.
    void cool(float dT);

    // Получить текущее давление в газгольдере.
    // Считать, что для газа верно уравнение состояния PV = (m/M)RT.
    // Значение постоянной R принять 8.31 Дж/(моль*К).
    float getPressure();
};
```

Внимание: в разных порциях могут быть впрыснуты разные газы с разной молярной массой. В этом случае общее давление газовой смеси в газгольдере, разумеется, является суммой парциальных давлений её компонентов.

Код для базового тестирования реализации класса:

```
    GasHolder h(1.0);
    h.inject(29, 29);
    cout << "Pressure after operation: " << h.getPressure() << " Pa" << endl;
    h.inject(29, 29);
    cout << "Pressure after operation: " << h.getPressure() << " Pa" << endl;
    h.heat(273);
    cout << "Pressure after operation: " << h.getPressure() << " Pa" << endl;
    h.cool(373);
    cout << "Pressure after operation: " << h.getPressure() << " Pa" << endl;
    h.cool(373);
    cout << "Pressure after operation: " << h.getPressure() << " Pa" << endl;
```

Базовый тест должен вывести:

```
Pressure after operation: 2268.63 Pa
Pressure after operation: 4537.26 Pa
Pressure after operation: 9074.52 Pa
Pressure after operation: 2875.26 Pa
```

```
Pressure after operation: 0 Pa
```

Космодром

Реализуйте класс космодрома со следующим прототипом:

```
class SpacePort
{
public:
    // Создать космодром, в котором size штук доков.
    // Доки нумеруются от 0 до size-1.
    // В момент создания все доки свободны.
    SpacePort(unsigned int size);

    // Запросить посадку в док с номером dockNumber.
    // Если такого дока нет - вернуть false (запрет посадки).
    // Если док есть, но занят - вернуть false (запрет посадки).
    // Если док есть и свободен - вернуть true (разрешение посадки) и док
становится занят.
    bool requestLanding(unsigned int dockNumber);

    // Запросить взлёт из дока с номером dockNumber.
    // Если такого дока нет - вернуть false (запрет взлёта).
    // Если док есть, но там пусто - вернуть false (запрет взлёта).
    // Если док есть и в нём кто-то стоит - вернуть true (разрешение взлёта) и
док становится свободен.
    bool requestTakeoff(unsigned int dockNumber)
};
```

Код для базового тестирования реализации класса:

```
    SpacePort s(5);
    cout << boolalpha << s.requestLanding(0) << endl;
    cout << boolalpha << s.requestLanding(4) << endl;
    cout << boolalpha << s.requestLanding(5) << endl;

    cout << boolalpha << s.requestTakeoff(0) << endl;
    cout << boolalpha << s.requestTakeoff(4) << endl;
    cout << boolalpha << s.requestTakeoff(5) << endl;
```

Базовый тест должен вывести:

```
true
true
false
true
true
false
```