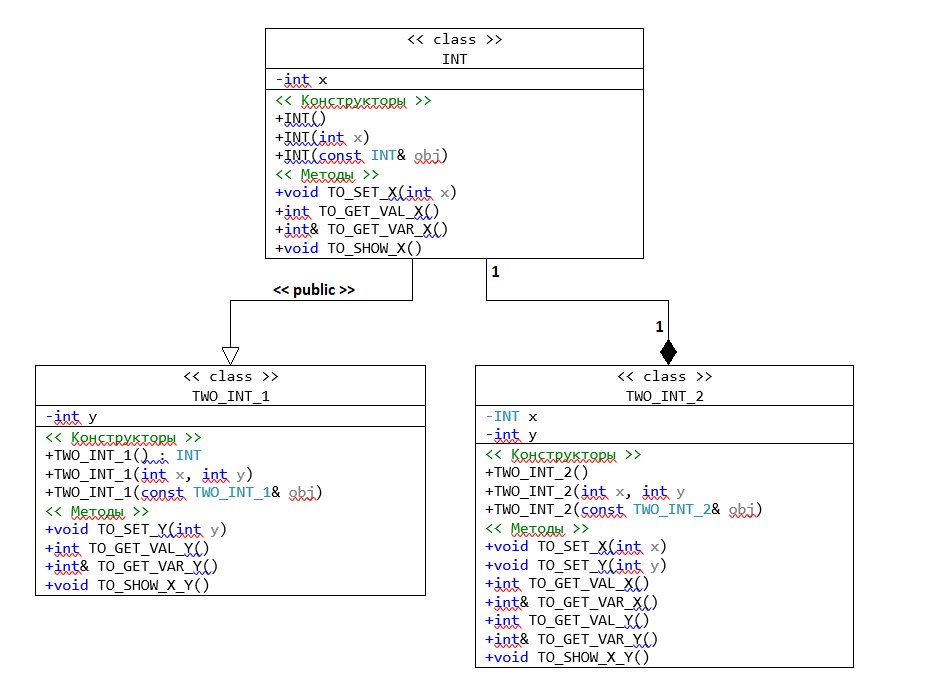
**ТАБЛИЦА UML**

****

**MAIN и ВЫВОД**

|  |  |
| --- | --- |
| int main()  {  INT i(23);  i.TO\_SHOW\_X();  i.TO\_SET\_X(32);  i.TO\_SHOW\_X();  TWO\_INT\_1 two\_i1(12, 21);  two\_i1.TO\_SHOW\_X\_Y();  two\_i1.TO\_SET\_X(1);  two\_i1.TO\_SET\_Y(2);  two\_i1.TO\_SHOW\_X\_Y();  TWO\_INT\_2 two\_i2(34, 55);  two\_i2.TO\_SHOW\_X\_Y();  two\_i2.TO\_SET\_X(3);  two\_i2.TO\_SET\_Y(5);  two\_i2.TO\_SHOW\_X\_Y();  } |  |

**РЕАЛИЗАЦИИ КЛАССОВ**

|  |
| --- |
| **INT** |
| class INT {  private:  int x;  public:  INT() : x(0) {}  INT(int x) : x(x) {}  INT(const INT& obj) : x(obj.x) {}  void TO\_SET\_X(int x) { this->x = x; }  int TO\_GET\_VAL\_X() { return x; }  int& TO\_GET\_VAR\_X() { return x; }  void TO\_SHOW\_X() { cout << x << endl; }  }; |
| **TWO\_INT\_1** |
| class TWO\_INT\_1 : public INT {  private:  int y;  public:  TWO\_INT\_1() : INT() { y = 0; }  TWO\_INT\_1(int x, int y) : INT(x), y(y) {}  TWO\_INT\_1(const TWO\_INT\_1& obj) : INT(obj), y(obj.y) {}  void TO\_SET\_Y(int y) { this->y = y; }  int TO\_GET\_VAL\_Y() { return y; }  int& TO\_GET\_VAR\_Y() { return y; }  void TO\_SHOW\_X\_Y() { cout << TO\_GET\_VAL\_X() << ' ' << y << endl; }  }; |
| **TWO\_INT\_2** |
| class TWO\_INT\_2 {  private:  INT x;  int y;  public:  TWO\_INT\_2() { x.TO\_SET\_X(0); y = 0; }  TWO\_INT\_2(int x, int y) : y(y) { this->x.TO\_SET\_X(x); }  TWO\_INT\_2(const TWO\_INT\_2& obj) : y(obj.y) { this->x.TO\_SET\_X(TO\_GET\_VAL\_X()); }  void TO\_SET\_X(int x) { this->x.TO\_SET\_X(x); }  void TO\_SET\_Y(int y) { this->y = y; }  int TO\_GET\_VAL\_X() { return x.TO\_GET\_VAL\_X(); }  int& TO\_GET\_VAR\_X() { return x.TO\_GET\_VAR\_X(); }  int TO\_GET\_VAL\_Y() { return y; }  int& TO\_GET\_VAR\_Y() { return y; }  void TO\_SHOW\_X\_Y() { cout << TO\_GET\_VAL\_X() << ' ' << y << endl; }  }; |

|  |
| --- |
| **CAR & TRACK** |
| class VEHICLE {  protected:  string number;  double speed;  double weight;  double acceleration;  public:  // Конструкторы  VEHICLE(string number, double weight, double acceleration);  // Методы  void GASS(double time);  void STOP(double time);  // Друзья  friend int COMPARE\_SPEED(const class CAR& obj1, const class TRACK& obj2);  };  // Конструкторы  VEHICLE::VEHICLE(string number, double weight, double acceleration) : speed(0.0) {  this->number = number;  this->weight = weight;  this->acceleration = acceleration;  }  // Методы  void VEHICLE::GASS(double time) { speed += acceleration \* time; }  void VEHICLE::STOP(double time) { speed -= acceleration \* time; if (speed < 0) speed = 0; }  class CAR : public VEHICLE {  public:  // Конструкторы  CAR(string number, double weight, double acceleration);  };  // Конструкторы  CAR::CAR(string number, double weight, double acceleration) : VEHICLE(number, weight, acceleration) {}  class TRACK : public VEHICLE {  private:  double track\_weight;  double trailer\_weight;  public:  // Конструкторы  TRACK(string number, double track\_weight, double trailer\_weight, double acceleration);  };  // Конструкторы  TRACK::TRACK(string number, double track\_w, double trailer\_w, double acceleration) : VEHICLE(number, weight, acceleration) {  track\_weight = track\_w;  trailer\_weight = trailer\_w;  }  // Друзья  int COMPARE\_SPEED(const CAR& obj1, const TRACK& obj2) {  if (obj1.speed > obj2.speed) {  cout << "The speed(" << obj1.speed <<  ") of car(" << obj1.number << ") is more than track's(" << obj2.number <<  ") speed(" << obj2.speed << ")." << endl;  return 1;  }  if (obj1.speed < obj2.speed) {  cout << "The track's(" << obj2.number << ") speed(" << obj2.speed <<  ") is more than speed(" << obj1.speed <<  ") of car(" << obj1.number << ")." << endl;  return 2;  }  cout << "The speeds(" << obj1.speed << ") of this vehicles(" << obj1.number <<  ' ' << obj2.number << ") are equal." << endl;  return 0;  } |