# Объявление объектов для таблиц (контейнеры и т.д.)

* Таблица констант (Номер константы, тип константы, значение).

class ConstTableItem{  
private:  
  
  
 ConstType\* constType;  
 string utf8;  
 double floatVal;  
 int val1, val2;  
  
public:  
 //методы  
};  
  
class ConstTable{  
private:  
 vector<ConstTableItem\*> items;  
public:  
 //методы на добалвения получение удаление, обновление  
};

* Таблица полей класса. Таблица хранит поля struct и константы, хранящиеся внутри trait.
* class FieldTableItem{  
  private:  
   TypeNode\* type;   
   bool isConst;   
   string id;   
   ExprNode\* value;   
  public:  
   //методы  
  };  
    
  class FieldTable{  
  private:  
   vector<FieldTableItem\*> items;  
  public:  
   //методы на добалвения получение удаление, обновление  
  };
* Таблица локальных переменных.

class VarTableItem{  
private:  
 TypeNode\* type;  
 bool isRef; // фактически это можно засунуть в тип после преобразования дерева  
 bool isMut;  
 string id;  
 ExprNode\* value;  
public:  
 //методы  
};  
  
class VarTable{  
private:  
 vector<VarTableItem\*> items;  
public:  
 //методы на добалвения получение удаление, обновление  
};

* Таблица методов. Свободные функции будут представлять собой статические методы анонимного класса с именем MY\_RUST\_ANONIMUS\_CLASS
* class MethodTableItem{  
  private:  
   string name;  
   TypeNode\* returnType;  
   ExprNode\* body;  
   MethodType\* methodType; // static, not\_static  
   FuncParamListNode\* params;  
   VarTable\* varTable;  
    
  public:  
   //методы  
  };  
    
  class MethodTable{  
  private:  
   string className;  
   map<string,MethodTableItem\*> items;  
  public:  
   //методы на добалвения получение удаление, обновление  
  };
* Таблица классов
* class ClassTableItem{  
  private:  
    
    
   ClassType classType; // stuct trait. можно разделить на классы  
   string name;  
   MethodTable\* methodTable;  
   FieldTable\* fieldTable;  
   ClassTableItem\* parent;  
    
  public:  
   //методы  
  };  
    
  class ConstTable{  
  private:  
   map<string,ClassTableItem\*> items;  
  public:  
   //методы на добалвения получение удаление, обновление  
  };

**Список ошибок**

1. Не определена fn main функция
2. Определено две и боее main функции
3. Несоответствие типов при инициализации
4. Несопоставимые типы при присвоениии
5. Несоответствие типов при передаче в функцию
6. Ожидаемый тип не совпадает с полученным типом 0308
7. Указана недопустимая область видимости (self/super)
8. Операнды имеют неприводимые типы
9. Недопустимое левое выражение в операции присваивания
10. Невозможно вычислить значение. (деление на ноль или целочисленное переполнение)
11. В enum два и более элемента имеют одно дискриминантное значение
12. Одинаковое имя полей у структуры
13. Возврат return из функции чей возвращаемый тип не ()
14. Бесконечный размер при определении рекурсивной структуры
15. Каждое поле при инициализации struct может быть проинициализировано лишь один раз
16. Не все поля при инициализации структуры были указаны.
17. Для диапазона range используется что-то другое кроме чисел и символов
18. Начало диапазона должно быть меньше или равно концу диапазона range
19. Ошибка вызова метода. Два и более метода имеют одинаковый прототип.
20. Попытка определить константу в trait
21. Конфликт реализаций (impl) одного trait для одно й struct
22. Метод не имеет параметров (у метода всегда должен быть хотя бы &self)
23. Попытка вызвать деструктор на struct вручную (через .drop())
24. Не найдены реализации в impl элементов, определённых в trait
25. Реализация в impl элемента с одним идентификатором из trait два и более раз
26. Попытка реализовать impl не для struct
27. Несоответствующие реализации в impl элементов, определённых в trait
28. Недопустимое кол-во аргументов при вызове функции
29. Несколько параметров функции имеет одно и то же имя
30. Значения для statics и constants должны быть известны к компиляции
31. Попытка изменить const или static
32. Вызов non-const функции при инициализации const
33. Обращение к необъявленному имени struct
34. Обращение к необъявленному имени module
35. Обращение к необъявленному в impl методу struct
36. Обращение к необъявленной свободной функции
37. Обращение к необъявленному локальной переменной
38. Обращение к необъявленному полю struct
39. Обращение к необъявленному имени trait
40. Использование не проинициализированной переменной
41. Использование в теле функции self когда он не передается в параметрах
42. Использование переменной после того как ее содержимое было перемещено 0382
43. Цикл зависимости типов 0391
44. Неразрешенный импорт (для модуля)
45. Неразрешенное имя (идентификатор совпадает с зарезервированными именами)
46. Struct или Module Определен более одного раза.
47. Использование ключевого слова pub там где оно не разрешено