Assigners and convertors.

Напомню что мы уже решили перегружать операцию := для удобства записи в атрибуты юнитов.

**abstract unit** Any

:= (that: **as this**) **is external**

**end**

**val unit** Real

:= (other: Integer) **is external**

**end**

r: Real **is** 5

r := 6

Таким образом наличие в юните Real перегруженной операции := позволяет присваивать сущностям типа Real не только сущности типа Real

Но возникает и обратный вопрос и сейчас нам надо писать так

i1 **is** Integer (5.5)

Таким образом, внутри юнита должны быть процедуры инициализации, которые могли бы построить валидный объект из сущностей других типов. Но это ограничивает возможности преобразования только за счет тех инициализаторов, которые прописаны внутри юнита. А хотелось бы написать так

i2: Integer is 5.5

или передать Real функции которая ожидает тип Integer. Т.е. хотелось бы написать конвертор в том типе который передается, а не в том который получает. Так как опять семантика связана с присваиванием, то и синтаксис можно рассмотреть аналогичный, но если в случае с присваиванием самому объекту у процедуры := должен быть один параметр, то в случае обратном (преобразование в некоторый тип) := должно быть функцией, которая этот нужный тип и возвращает.

**val unit** Real

:= (): Integer **is** …. **end**

**end**

таким образом в любом месте при передаче параметров надо проверять, а не присутствует ли функция преобразования …