004: renSigdoc (A.Tycon n) _ がハードコーディングされているのを直す

↑ up

- issued: 2020-04-07
- 分類：C 改善項目
- status: Closed (2020-04-08)

現象

限られた型コンストラクタに関する変換だけがハードコーディングされている。

```
-- TODO: should be fix this hard coding.
renSigdoc (A.Tycon n) _ = case origName n of
  "Integer" -> return tInteger
  "Int"     -> return tInt
  "String"  -> return tString
  "IO"      -> return $ TCon (Tycon "IO" (Kfun Star Star))
  "()"      -> return tUnit
  "Bool"    -> return tBool
  s         -> error $ "renSigDoc $ A.Tycon " ++ s
```

調査ログ

2020-04-08 (Wed)

`scandecl d@(A.DataDecl (maybe_context, ty) consts maybe_dtys)` 内の `renTy (A.Tycon i)` にも同様の問題がある。

```
renTy (A.Tycon i) = case origName i of  -- TODO:
  "Int"  -> tInt
  "Integer" -> tInteger
  "Char" -> tChar
  x -> error $ "Non-exhaustive patterns: " ++ x
```

現状の実装では、たとえばデータ構築子の型は、`scandecl d@(A.DataDecl (maybe_context, ty) consts maybe_dtys)` の中で環境に辞書（`Assump`) として保持されているが、型構築子については、同様のものがなく、こうして決め打ちで解決している状態。

型構築子についても、引き回す必要がある。(引き回す、というのは、分割コンパイルを想定)

まずは、いまの決め打ちと同等の辞書を用いるようにして、つぎに、データ宣言でつくられたやつを処理(これをやると adt-sample4.hs が通るはず)。

モジュールの import/export をきちんと処理するのは、他の Assump も一緒にやる必要があるので、あとまわしにしたい。後回しにするタスクは以下:

- 型構築子の Assump が export/import されるようにする
- qualified 識別子を適切に処理する

これらを後回しにしつつ、現状と同等の処理を決め打ちじゃなくすことのみ実施。

⇒ 前項は、RnState に辞書を付け加えたことで自然に達成された。後者を 005 として採番。

今回は、現状のままリファクタリングしたので、TypeConsts の型は [(Id, Type)] としたが、[Assump] でなくていいのだろうか (006)