056: retint.hs で cannot resolve ambiguity

↑ up

- issued: 2020-05-04
- 分類： A サンプルコードが fail
- status: Closed (2020-05-06)

概要

retint.hs をコンパイルすると型推論で失敗、cannot resolve ambiguity となる。
retint.hs:

```
strlen ""     = 0
strlen (c:cs) = 1 + strlen cs

main = return (strlen "Haskell")
```

調査ログ

2020-05-06 (Wed)

このエラーは、Typing で発生している（ので、後段の DictPass などの問題ではない）。
たしかに、なんの型注釈もないと、return が何モナドのインスタンスかわからないだろう。

- 型注釈なし → errror: cannot resolve ambiguity
- main :: IO a とする → error: context too weak
- main :: IO Int → OK
- main :: IO () → error: context reduction

main に型注釈がない場合には、以下のように最後に >> return () を追加しつつ、型注釈 main :: IO () をつければ、仕様（main は IO t を計算して最後に t をすてる）が満たせるのでは。

```
strlen ""     = 0
strlen (c:cs) = 1 + strlen cs

main :: IO ()
main = do { return (strlen "Haskell") } >> return ()
```

なお、main :: [Int] のように IO t でない型注釈を明に書いてもはじかれるので、そういうチェックは必要。
(main :: ty) main = e
を
main :: IO () main = e >> return ()
または、
main :: IO () main = (e :: ty) >> return ()
に変形してやれば、ty が IO t の形をしていることのチェックも含め実現できそう。
⇒ 変形しなくても、型検査器に制約をしらせるだけであれば、以下のように特別な関数 main' を追加してやるだけで OK っぽい。

```
strlen ""     = 0
strlen (c:cs) = 1 + strlen cs

main = return (strlen "Haskell")

main' :: IO ()
main' = main >> return ()
```

このようなダミー宣言を Semant で付け加えるようにして、期待通りに動作した。
retint.hs は sample179.hs へ。