

Proc/fun-decls in Pascal

Pascal prog:

```

program Foo;
  type;
  var;
  function ----
  procedure ----
  proc ----
begin
...
end.

```

func-decl:

```

function heading semi;
any-decl-part
statement-part semi;

function Bar (....) : integer;
  type --
  var --
  function aux (....) .
  begin
  ...
end;
begin
...
end;

```

```

Function FtoC (fahrheit: Real):
  Real;
begin
  FtoC := (fahrheit - 32) * 5/9;
end;

```

Function-declaration:

function-heading semi
 (look up fun name in symbol
 if new, install as PROC
 else if ~~PROC~~ a function type
 and type of fun match &
 not external, then
 change PROC to PROC
 Save fun name (to recognise
 when an assignment is setting
 the return value)
 stack-block()
 Install formal parameters in the
 symbol table:
 b.install-formal-parameters().
 for each parameter (left to right)
 install param in symbol as PROC
 with offset determined by
 adding
 = b.get-formal-parameters() +
 offset
 (if var parameters
 should be 4*PTR)

save get-local-var-offset()
 does not this offset
 by 4 if needed
 return type
 {makes room for
 return accumulator}

and -16, skip 3

any-decl-part "main"
 { b.fun-prologue (fun name)
 "store" each formal param:
 for each formal param in order
 offset = b.get-formal-param (name)
 nice check: this offset
 should be the same as
 what you got before
 (if not, long (...)
 if nonvoid return ["Function"]
 b.alloc-return-value();
 b.alloc-var (local var names)
 (statement-part) semi;
 { b.prepare-return ();
 b.fun-epilogue (fun name)
 if done
 block
 stack-block() }

Mar 29-11:03 AM