```
/* Functions */

/*
 * Functions types
 */
mut f i32 = 55
funcType type = fn (i32, u32) i32

#fnType funcType
fn addIU (x i32, y u32) i32 {
  ret x + y
}

/*
 * functions can have multiple return values
 * NOTE: this is greate to return error codes!
 */
fn divide (x i32, y u32) i32, error #must {
  if y == 0 {
    ret 0, error(-1, “Division by zero”)
  }
  ret x + y, nil
}


fn myProgram () i32 {
  mut x i32; mut y i32 = 8
  c i32 = 100

   x = y
   y = f
   mut f := c * (x + y) % 62

   [x, y] captureXY {
     /* local scope that captures outer variables */
     /* name is optional just to better reading */
   }

   namedScope {
     /* local scope to reduce procedure namespace overhead */
     /* name is optional just to better reading */
     /* every outer variable is available */
   }

   ret f
}

#main
fn main (args string[]) i32 {
  ret myProgram()
}
```