Problem 4

Find the largest palindrome made from the product of two 3-digit numbers.

Let our palindrome be $P = ab$ with a and b the two 3-digit numbers. If a and b are 3-digits long then they must lie between 100 and 999 inclusive. So an initial solution to the problem might be:

```
function reverse(n)
     reversed = 0
     while n > 0
          reversed = 10*reversed + n mod 10
          n = n/10
     return reversed

function isPalindrome(n)
     return n = reverse(n)

largestPalindrome = 0
a = 100
while a <= 999
     b = 100
     while b <= 999
          if isPalindrome(a*b) and a*b > largestPalindrome
               largestPalindrome = a*b

          b = b+1
     a = a+1

output largestPalindrome
```

This is fast enough for this case but could be improved. For starters, the current method checks many numbers multiple times. For example the number 69696 is checked both when a=132 and b=528 and when a=528 and b=132. To stop checking numbers like this we can assume $a \leq b$, roughly halving the number of calculations needed.

This would change the code as follows:

```
//…

largestPalindrome = 0
a = 100
while a <= 999
     b = a //Now b=a instead of 100
     while b <= 999
          if isPalindrome(a*b) and a*b > largestPalindrome
               largestPalindrome = a*b

          b = b+1
     a = a+1

output largestPalindrome
```

Next we should consider counting *downwards* from 999 instead of counting *upwards* from 100. This makes the program far more likely to find a large palindrome earlier, and we can quite easily stop checking *a* and *b* that would be too small to improve upon the largest palindrome found so far.

```
largestPalindrome = 0
a = 999
while a >= 100
     b = 999
     while b >= a
          if a*b <= largestPalindrome
               break //Since a*b is always going to be too small

          if isPalindrome(a*b)
               largestPalindrome = a*b

          b = b-1
     a = a-1

output largestPalindrome
```

This is fast but can be sped up further with some analysis. Consider the digits of *P* – let them be *x*, *y* and *z*. *P* must be at least 6 digits long since the palindrome 111111 = 143×777 – the product of two 3-digit integers. Since *P* is palindromic:

$$P = 100000x + 10000y + 1000z + 100z + 10y + x$$
$$P = 100001x + 10010y + 1100z$$
$$P = 11(9091x + 910y + 100z)$$

Since 11 is prime, at least one of the integers *a* or *b* must have a factor of 11. So if *a* is not divisible by 11 then we know *b* must be. Using this information we can determine what values of *b* we check depending on a:

```
largestPalindrome = 0
a = 999
while a >= 100
     if a mod 11 = 0
          b = 999
          db = 1
     else
          b = 990 //The largest number less than or equal 999
                  //and divisible by 11
          db = 11

     while b >= a
          if a*b <= largestPalindrome
               break

          if isPalindrome(a*b)
               largestPalindrome = a*b

          b = b-db
     a = a-1

output largestPalindrome
```