# 3 Impossible Programs

(a) Cannot exist

We proceed with a proof by contradiction and use the reduction technique. Suppose, for a contradiction, that such a program $P$ exists. Using the given information, so the program looks like this:

$P(F, x, y)$
  if $F(x) = y$, then return True
  else, return False

Thus, this function could be used as a subroutine to solve the Halting Problem where we construct an algorithm like this:

Halt$(Program, x)$
  Construct a program $Program'$ that, on any input, returns $Program(x)$
  return $P(Program', x, Program(x))$

where $Program'$ can be constructed rather easily (following Note 11):

$Program'(y)$
  return $Program(x)$

So, we can see that $Program'(x)$ returns $Program(x)$ if and only if $Program(x)$ halts. Thus, by assumption of the program $P$ in the problem, so $P(Program', x, Program(x)$ is True if and only if $Program(x)$ halts.

Therefore, if we have such a program $P$, then Halt will correctly solve the Halting Problem. Since we know there cannot be such a program Halt (the Halting Problem is uncomputable), so we conclude with contradiction, which means that this program $P$ does not exist.

Q.E.D.

(b) Cannot exist

We proceed with a proof by contradiction and use the reduction technique. Suppose, for a contradiction, that such a program $P$ exists. Using the given information, so the program looks like this:

$P(F, G)$

If for all $x$, either both $F(x)$ and $G(x)$ halts or both $F(x)$ and $G(x)$ loops, then return True
else, return False

Thus, this function could be used as a subroutine to solve the Halting Problem where we construct an algorithm like this:

Halt($Program, x$)

Construct a program, $TestyHalt$, that halts on input $x$ (i.e. returns 0 directly on input $x$), and returns $Program(x)$ otherwise
return $P(Program, TestyHalt)$

where $TestyHalt$ can be constructed rather easily:

$TestyHalt(y)$

if $y == x$, then return 0 (halts)
else, return $Program(x)$

So, we can see that $Program$ and $TestyHalt$ are constructued to halt on the same inputs for all inputs except $x$, which is the only input we would need to examine. Then, since $TestyHalt$ always halts on input $x$, so $P(Program, TestyHalt)$ is True if and only if $Program(x)$ halts just like $TestyHalt(x)$ does.

Therefore, if we have such a program $P$, then Halt will correctly solve the Halting Problem. Since we know there cannot be such a program Halt (the Halting Problem is uncomputable), so we conclude with contradiction, which means that this program $P$ does not exist.

Q.E.D.