Let $W = \{ \langle P \rangle \mid P \text{ on input 0 eventually halts} \}$.

**Claim** Given an algorithm $A_W$ to compute (decide) $W$, there is an algorithm to compute $A_{\text{Prog}}$.

**Proof** Here is the algorithm deciding $A_{\text{Prog}}$.

On input $\langle P, w \rangle$:

We need to build (compute) a program $Q_{P,w}$ which we then input to the algorithm $A_W$.

What we want is for $A_W(\langle Q_{P,w} \rangle) = \begin{cases} 
\text{"accept"} & \text{if } P \text{ eventually halts on input } w \\
\text{"reject"} & \text{if } P \text{ does not halt on input } w
\end{cases}$

This means that $Q_{P,w}(0)$ halts if $P$ eventually halts on input $w$ and does not halt if $P$ does not halt on input $w$ for any $x$, and hence it does so for $x = 0$ in particular.

Here is a program $Q_{P,w}$ that meets the above requirement:

On input $x$:

$Q_{P,w}$ runs $P$ on input $w$ (i.e. $Q_{P,w}$ ignores its input).

Clearly $Q_{P,w}(0)$ halts if $P$ eventually halts on input $w$ and does not halt if $P$ does not halt on input $w$

for any $x$, and hence it does so for $x = 0$ in particular.

Thus our algorithm for deciding $A_{\text{Prog}}$ proceeds as follows:

1. Form $\langle Q_{P,w} \rangle$, the encoding of program $Q_{P,w}$.
2. Run algorithm $A_W$ on input $\langle Q_{P,w} \rangle$ and use the result of $A_W(\langle Q_{P,w} \rangle)$ as the answer to report.

\[ \square \]

Let $x = \{ \langle P \rangle \mid \text{either } P \text{ on input 0 eventually halts, or } P \text{ on input 1 eventually halts, or both} \}$.

**Claim** Given an algorithm $A_X$ to decide $X$, there is an algorithm to decide $A_{\text{Prog}}$.
Proof  This is very similar to the previous algorithm. Again, we need to build a program $Q'_{P,w}$ with the following characteristics:

\[
A_X((Q'_{P,w})) \text{ halts if } P \text{ eventually halts on input } w \\
\text{and does not halt if } P \text{ does not halt on input } w
\]

This means that

- if $P$ eventually halts on input $w$: at least one of $Q'_{P,w}((0))$ and $Q'_{P,w}((1))$ halts
- if $P$ does not halt on input $w$: both $Q'_{P,w}((0))$ and $Q'_{P,w}((1))$ do not halt.

It is easy to check that $Q'_{P,w} = Q_{P,w}$ meets this requirement. Thus we can use the following algorithm to decide $A_{prog}$.

On input $(P, w)$:
1. Construct the encoding $(Q_{P,w})$.
2. Run $A_X((Q_{P,w}))$ and give its result as the output for the algorithm.

□

Let $Y = \{P \mid$ there is a variable $v$ in $P$ that is never assigned a value when run on input 1 $\}$.

Claim  Given an algorithm $A_Y$ to decide $Y$, there is an algorithm to compute $A_{Prog}$.

Proof  Here is the algorithm to decide $A_{Prog}$.

On input $(P, w)$:

Our algorithm will need to construct a program $Q''_{P,w}$ which we then input to $A_Y$. What we want is for:

\[
A_Y((Q''_{P,w})) \text{ halts if } P \text{ eventually halts on input } w \\
\text{and does not halt if } P \text{ does not halt on input } w
\]

This means that:

- If $P$ eventually halts on input $w$ then $Q''_{P,w}$ has a variable that is never assigned a value when run on input 1, and
- if $P$ does not halt on input $w$ then every variable of $Q''_{P,w}$, when run on input 1, is assigned a value.

Let’s try $Q''_{P,w}$ is the program:

On input $x$:
1. Run $P((w))$
2. For every variable $z$ appearing in $P$ do: $z \leftarrow 0$
3. \( v \leftarrow 1 \), where \( v \) is a variable that does not appear in \( P \)

We see that if \( P \) halts on input \( w \), then every variable appearing in \( Q_{P,w}^{\prime} \) is assigned a value when \( Q_{P,w}^{\prime} \) is run on input \( x = 1 \), while if \( P \) does not halt on input \( w \), at the very least, variable \( v \) in \( Q_{P,w}^{\prime} \) is not assigned a value.

Oops, this is back to front. Unfortunately, this is unavoidable. So let’s change our goal for \( A_Y \).

Let’s require

\[
A_Y(\langle Q_{P,w}^{\prime} \rangle) \text{ does not halt if } P \text{ eventually halts on input } w \\
\text{ and halts if } P \text{ does not halt on input } w
\]

Now, our algorithm for deciding \( A_{\text{Prog}} \) simply reports the opposite answer to \( A_Y(\langle Q_{P,w}^{\prime} \rangle) \).

So the algorithm is the following:

1. Construct the encoding \( \langle Q_{P,w}^{\prime} \rangle \).
2. Run \( A_Y(\langle Q_{P,w}^{\prime} \rangle) \).
3. Report the opposite answer to that given by \( A_Y \) in Step 2.

\( \Box \)