Let $A$ be a countable set. $A = \{a_1, a_2, a_3, \ldots\}$

By $(1)$, the set of all finite subsets of $A$ is countable.

The set $A$ is countable, and so the set of finite subsets of

A finite subset of $A$ is a subset of a countable set.

We now consider the set of countable sets, $A = \{B_1, B_2, B_3, \ldots\}$

We form a new countable list by diagonalizing over all countable elements $A$. The new list is $A$, where $A = \{a_1, a_2, a_3, \ldots\}$

Because of the construction of $A$, any countable subset of $A$ contains every element of $A$.

So $A$ is countable, and so the set of finite subsets of $A$ is countable.
Consider the program
\[ I_1 : J(1, 2, 6) \quad I_2 : S(2) \quad I_3 : S(3) \quad I_4 : J(1, 2, 6) \quad I_5 : (1, 1, 2) \quad I_6 : T(3, 1) \]
Show that the computation under this program with initial configuration 2, 3, 0, 0, ... never stops.

In order for this URM to stop, it has to execute \( I_6 : T(3, 1) \) (cf. Fig.1b on p.15). And since this is an arithmetic instruction, the URM stops whenever it reads \( I_6 \).

In order to proceed to \( I_6 \), it has to execute \( I_1 : J(1, 2, 6) \) or \( I_4 : J(1, 2, 6) \) with \( r_1 = r_2 \). This is because \( I_5 : J(1, 1, 2) \) creates a loop and never let the URM to proceed to \( I_6 \).

In our initial configuration, \( r_1 = 2, r_2 = 3 \) and \( r_1 < r_2 \). Hence \( r_1 \neq r_2 \). Since the only instructions besides \( I_1 \) and \( I_4 \) that occur in the loop are \( I_2 : S(2), I_3 : S(3) \), it is always the case that \( r_1 < r_2 \).
and hence \( r_1 \neq r_2 \). However, in order to proceed to \( I_6 \) through either of the jump instructions \( I_1 \) and \( I_4 \), it has to be the case that \( r_1 = r_2 \). Thus, the URM with this initial configuration does not stop.
Begin with the configuration \(x, y, 0, 0, 0, \ldots\)

<table>
<thead>
<tr>
<th>Step</th>
<th>Condition</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(i, j \in {1, 2, 3, 4, 5})</td>
<td>Checks if (x) or (y) is (0)</td>
</tr>
<tr>
<td>2</td>
<td>(i &gt; j)</td>
<td>(x = y) counts (x) or (y)</td>
</tr>
<tr>
<td>3</td>
<td>(i &gt; j)</td>
<td>Steps (x-y) counts (x) or (y)</td>
</tr>
<tr>
<td>4</td>
<td>(i &gt; j)</td>
<td>Steps (x)-counter (x) for (i) and (j)</td>
</tr>
<tr>
<td>5</td>
<td>(i &gt; j)</td>
<td>(y)-counter (y) for (i) and (j)</td>
</tr>
<tr>
<td>6</td>
<td>(i &gt; j)</td>
<td>(z)-counter (z) for (i) and (j)</td>
</tr>
<tr>
<td>7</td>
<td>(i &gt; j)</td>
<td>(x)-counter (x) for (i) and (j)</td>
</tr>
<tr>
<td>8</td>
<td>(i &gt; j)</td>
<td>(y)-counter (y) for (i) and (j)</td>
</tr>
<tr>
<td>9</td>
<td>(i &gt; j)</td>
<td>(z)-counter (z) for (i) and (j)</td>
</tr>
<tr>
<td>10</td>
<td>(i &gt; j)</td>
<td>(x)-counter (x) for (i) and (j)</td>
</tr>
<tr>
<td>11</td>
<td>(i &gt; j)</td>
<td>(y)-counter (y) for (i) and (j)</td>
</tr>
<tr>
<td>12</td>
<td>(i &gt; j)</td>
<td>(z)-counter (z) for (i) and (j)</td>
</tr>
</tbody>
</table>

\(x, y\) is stored in \(R_1\). \(f(x, y) = xy\)
Exercise 4

Theorem 2. If $P$ is a program without jump, $\exists m \in \mathbb{N}$ such that $\forall x (f_{x,1}(x) = m) \lor \forall x (f_{x,1}(x) = x + m)$

Let us prove theorem 2 by induction on the length of $P$. The initial state of the registers is $x, 0, 0, \ldots$. We define $P(P)(s)$ as:

Definition: $P(P)(s)$ is $P$ has no jump and $P$ is of length $s$ entails that $\exists m \in \mathbb{N}$ such that $\forall x (f_{x,1}(x) = m) \lor \forall x (f_{x,1}(x) = x + m)$

Basis (B): $P(P)(1)$
Induction step (I): $\mathcal{P}(P(s)) \rightarrow \mathcal{P}(P(s+1))$

Proof of B: Since there is no jump instruction in $P$, then $P$ contains only instructions of the form $Z(u)$, $S(u)$ or $T(u,v)$, for some $u,v \leq \rho(P)$. By hypothesis, there is only one instruction in $P$. Then:

- If this instruction is of the form $Z(n)$, $S(n)$ or $T(m,n)$ for some $m,n \in \mathbb{N}$ and $n > 1$ and $m \geq 1$, then the contents of $R_1$ is not affected by $P$, and
  \[ \forall x : P(x) \downarrow x. \] Then $\exists m \in \mathbb{N}$ such that $\forall x(f^P_1(x) = x + m)$, with $m = 0$.

- If this instruction is of the form $Z(1)$ or $T(m,1)$ for some $m \in \mathbb{N}$ and $m > 1$, then $\forall x : P(x) \downarrow 0$ and $\exists m \in \mathbb{N}$ such that $\forall x(f^P_1(x) = m)$, with $m = 0$.

- If this instruction is of the form $T(1,1)$ then $\forall x : P(x) \downarrow x$ and $\exists m \in \mathbb{N}$ such that $\forall x(f^P_1(x) = x + m)$, with $m = 0$.

- If this instruction is of the form $S(1)$ then $\forall x : P(x) \downarrow x + 1$ and $\exists m \in \mathbb{N}$ such that $\forall x(f^P_1(x) = x + m)$, with $m = 1$.

There are no other possible instructions in a program $P$ of length 1 without jump.

In all cases, given a program $P$ of length 1 without jump, $\exists m \in \mathbb{N}$ such that $\forall x(f^P_1(x) = m) \lor \forall x(f^P_1(x) = x + m)$.

Proof of I: Let $P$ be of length $s$. If $\mathcal{P}(P(s))$, then for all $P$, $\exists m \in \mathbb{N}$ such that $\forall x, R_1 = x + m$ after the execution of $I_s$, or $R_1 = m$ after the execution of $I_s$. Let define the program $P_s$ as the concatenation of one instruction $I_{s+1}$ to $P$, such that $I_{s+1}$ is not a jump. Then:

- If $I_{s+1}$ is of the form $Z(n)$, $S(n)$ or $T(m,n)$ for some $m,n \in \mathbb{N}$ and $n > 1$ and $m \geq 1$, then the contents of $R_1$ is not affected by $I_{s+1}$, and for all $x$, $P(x) \downarrow b \rightarrow P_s(x) \downarrow b$. Given that $\mathcal{P}(P(s))$, this entails that $\exists m \in \mathbb{N}$ such that $\forall x(f^P_1(x) = m) \lor \forall x(f^P_1(x) = x + m)$

- If $I_{s+1}$ is of the form $S(1)$, then $R_1 = b + 1$ after the execution of $I_{s+1}$, where $b$ is the contents of $R_1$ right after the execution of $I_s$. Since $\mathcal{P}(P(s))$, $\exists m \in \mathbb{N}$ such that $\forall x(f^P_1(x) = m) \lor \forall x(f^P_1(x) = x + m)$

- If $I_{s+1}$ is of the form $Z(1)$ or $T(x,1)$, where $x > \rho(P)$, then $R_1 = 0$ after the execution of $I_{s+1}$. Then $\exists m \in \mathbb{N}$ such that $\forall x(f^P_1(x) = m)$, with $m = 0$.

- If $I_{s+1}$ is of the form $T(v,1)$ for some $v$ such that $1 < v \leq \rho(P)$, then $\exists m \in \mathbb{N}$ such that $\forall x$ in the domain of $f^P_1$, either $R_1 = x + m$ right before the execution of $I_{s+1}$, or $R_1 = m$ right before the execution of $I_{s+1}$. This can be proven by contradiction with the induction hypothesis. Let $a_1$ and $a_2$ be two natural numbers in the domain of $f^P_1$, such that $a_1 \neq a_2$. Let us further assume that there exists two natural numbers $m$ and $n$ such that $m \neq n$ and $R_1 = a_1 + m$ right before the execution of $I_{s+1}$ if $R_1 = a_1$ in the initial state, but $R_1 = a_2 + n$ right before the execution of $I_{s+1}$ if $R_1 = a_2$ in the initial state. In this case, an application of $T(v,1)$ in $I_{s+1}$
would falsify theorem 2, hence there is a program of length $s + 1$ such that $P(P)(s + 1)$ is false. But then, since $1 < v \leq s$, there is also a program of length $s$ such that $I_v = T(v, 1)$, and $P(P)(s)$ is false. However, this contradicts our inductive hypothesis, which allows us to conclude that $\forall P$ of length $s$ without jump, $\forall v$ such that $1 < v \leq s$, $\exists m \in \mathbb{N}$ such that $\forall x$ in the domain of $f_{p_v}^x$, either $R_v = x + m$ right before the execution of $I_{s+1}$, or $R_v = m$ right before the execution of $I_{s+1}$. If $I_s = T(v, 1)$ in these conditions, $\exists m \in \mathbb{N}$ such that $\forall x (f_{p_v}^x(x) = m) \lor \forall x (f_{p_v}^x(x) = x + m)$.

- There is no other possible form of $I_{s+1}$ given that there are no jumps in $P^*$.

In all cases, $P(P)(s) \rightarrow P(P)(s + 1)$. Given that $P(P)(1)$ is true, we conclude by induction that for all $s \in \mathbb{N}$, $P(P)(s)$ is true.
7 For each instruction $T(m,n)$, there is a program $S$ such that every instruction with exactly the same effect as $T(m,n)$ ...

Suppose we have a program:

$I_1:$ ...
$I_2:$ ...
$\vdots$
$I_k : T(m,n)$
$I_{k+1}$:

We replace the $k^{th}$ instruction, $T(m,n)$, with the sub-program:

$I_{k+1} : Z(n)$
$I_{k+2} : J(m, n, k+1)$
$I_{k+3} : S(n)$
$I_{k+4} : J(1,1, \ldots k+1)$

This sub-program changes only the value of the register $P_n$, and makes reference only to the registers $P_r$ and $P_n$—so any other registers won't be affected.

And when the sub-program is done running (i.e., $m=m$), it jumps to the step that followed the original transfer instruction.