Sample Questions

**Question 1:**
Write a program \( P \) that computes \( \Psi_P(1)(r) = r - 2 \) using no macros.

**Notice** that we only consider natural numbers! This means that, for example, the expression \((1-2)\) is undefined. Any program computing such an expression must never halt.

**Sample solution:**

\[
\begin{align*}
X & \leftarrow X - 1 \\
\text{IF } X \neq 0 \text{ GOTO A2} \\
\end{align*}
\]

\[
\begin{align*}
A1 & \quad Z \leftarrow Z + 1 \\
\text{IF } Z \neq 0 \text{ GOTO A1} \\
\end{align*}
\]

\[
\begin{align*}
A2 & \quad X \leftarrow X - 1 \\
\quad Y \leftarrow Y + 1 \\
\text{IF } X \neq 0 \text{ GOTO A2} \\
\quad Y \leftarrow Y - 1
\end{align*}
\]

---

**Sample Questions**

**Question 2:**

a) Write a program \( S \) that computes \( \Psi_S(1)(r) = r - 4 \).

Use a macro based on program \( P \) of the form \( W \leftarrow p(V) \) (which has the effect \( W \leftarrow V - 2 \)).

**Sample solution:**

\[
\begin{align*}
Y & \leftarrow p(X) \\
Y & \leftarrow p(Y)
\end{align*}
\]

b) Now comes the best part: Expand all macros in \( S \) using the method we discussed in the lecture.

\[
\begin{align*}
Z_2 & \leftarrow 0 \quad \text{(expansion of } Y \leftarrow p(X) \text{ with } n = 2) \\
Z_1 & \leftarrow X \\
Z_0 & \leftarrow 0 \\
Z_{-1} & \leftarrow Z_1 \\
\text{IF } Z_0 = 0 \text{ GOTO A1} \\
Z_1 & \leftarrow Z_1 - 1 \\
\text{IF } Z_1 \neq 0 \text{ GOTO A1} \\
Z_0 & \leftarrow Z_0 - 1 \\
Y & \leftarrow Z_0 - 1 \\
\end{align*}
\]

---

**Composition**

Let us **combine** computable functions in such a way that the output of one becomes an input to another. For example, we could combine the functions \( f \) and \( g \) to obtain a new function \( h \): \( h(x) = f(g(x)) \).

Let us now take a more general view:

**Definition:** Let \( f \) be a function of \( k \) variables and let \( g_1, \ldots, g_k \) be functions of \( n \) variables. Let \( h(x_1, \ldots, x_k) = f(g_1(x_1, \ldots, x_n), \ldots, g_k(x_1, \ldots, x_n)) \).

Then \( h \) is said to be obtained from \( f \) and \( g_1, \ldots, g_k \) by composition.

---

**Theorem 1.1:** If \( h \) is obtained from the (partially) computable functions \( f, g_1, \ldots, g_k \) by composition, then \( h \) is (partially) computable.

**Proof:** The following program obviously computes \( h \):

\[
\begin{align*}
Z_1 & \leftarrow g_1(X_1, \ldots, X_n) \\
\vdots \\
Z_k & \leftarrow g_k(X_1, \ldots, X_n) \\
Y & \leftarrow f(Z_1, \ldots, Z_k)
\end{align*}
\]

If \( f, g_1, \ldots, g_k \) are not only partially computable but are also total, then so is \( h \).
Composition
Let us combine computable functions in such a way that the output of one becomes an input to another. For example, we could combine the functions $f$ and $g$ to obtain a new function $h$:

$$h(x) = f(g(x))$$

Let us now take a more general view:

**Definition:** Let $f$ be a function of $k$ variables and let $g_1, \ldots, g_k$ be functions of $n$ variables. Let

$$h(x_1, \ldots, x_n) = f(g_1(x_1, \ldots, x_n), \ldots, g_k(x_1, \ldots, x_n)).$$

Then $h$ is said to be obtained from $f$ and $g_1, \ldots, g_k$ by composition.

**Theorem 1.1:** If $h$ is obtained from the (partially) computable functions $f, g_1, \ldots, g_k$ by composition, then $h$ is (partially) computable.

**Proof:** The following program obviously computes $h$:

$$Z_1 \leftarrow g_1(X_1, \ldots, X_n)$$
$$\vdots$$
$$Z_k \leftarrow g_k(X_1, \ldots, X_n)$$
$$Y \leftarrow f(Z_1, \ldots, Z_k)$$

If $f, g_1, \ldots, g_k$ are not only partially computable but are also total, then so is $h$. ■