Ingeniería en Informática, UCM. Curso 2014-2015.
Calculabilidad y Complejidad, ejercicios a entregar, primera parte.

1. A continuación se muestran diversos modelos de cómputo:

   (I) Los programas *repeat* son los definidos a través de las siguientes reglas de formación de programas:

   $S ::= \; X_i := X_i + 1 \mid X_i := X_i \dot{-} 1 \mid X_i := 0 \mid$
   if $X_i = 0$ **then** S_1 **else** $S_2 \mid (S_1; S_2) \mid$
   repeat S_1 **until** $X_i = 0$

   donde el significado de la última construcción es ejecutar S_1 obligatoriamente una vez y, después, mientras X_i no valga 0, volver a ejecutar S_1.

   (II) Los programas for_fijo son los definidos a través de las siguientes reglas de formación de programas:

   $S ::= \; X_i := X_i + 1 \mid X_i := X_i \dot{-} 1 \mid X_i := 0 \mid$
   if $X_i = 0$ **then** S_1 **else** $S_2 \mid (S_1; S_2) \mid$
   for K **do** S_1

   donde el significado de la última construcción es ejecutar S_1 un número de veces igual a K, donde K es un número natural.

   (III) Los programas *como_mucho_cinco_whiles* son los definidos a través de las siguientes reglas de formación de programas:

   $S ::= \; X_i := X_i + 1 \mid X_i := X_i \dot{-} 1 \mid X_i := 0 \mid$
   if $X_i = 0$ **then** S_1 **else** $S_2 \mid (S_1; S_2) \mid$
   while $X_i \neq 0$ **do** $S_1 \mid$ **for** X_i **do** S_1

   donde el significado de la última construcción es ejecutar S_1 tantas veces como valga la variable X_i justo al llegar al bucle. **Además**, en cada programa *cinco_whiles*, como mucho pueden aparecer 5 instrucciones del tipo **while** $X_i \neq 0$ **do** S_1.

   Para cada uno de los modelos mostrados en (I), (II), (III), haz lo siguiente:

   (1) Si crees que la clase de funciones que se pueden computar con dicho modelo es la clase de las funciones recursivas parciales, (a) explica por qué los programas que podemos escribir en dicho modelo se pueden simular con programas *while* o con máquinas MRI (lo que prefieras); y (b) explica por qué los programas *while* o las máquinas MRI (lo que prefieras) se pueden simular con los programas de dicho modelo.

   (2) Si crees que la clase de funciones que se pueden computar con dicho modelo es la clase de las funciones recursivas primitivas, (a) explica por qué los programas que podemos escribir en dicho modelo se pueden simular con una función recursiva primitiva; y (b) explica por qué las funciones recursivas primitivas se pueden simular con los programas de dicho modelo.

   (3) Si crees que el tipo de funciones que se pueden computar con dicho modelo es otro, justifícalo e identifica con la mayor precisión que puedas cómo serán dichas funciones.

2. Dado un programa MRI P que recibe n números naturales como entrada, y dados $q < n$ números naturales $y_1, \ldots, y_q$, ¿existe algún método para crear automáticamente, a partir de P, un programa Q que reciba $n - q$ números naturales como entrada, de forma que se cumpla $Q(x_1, \ldots, x_{n-q}) = P(y_1, \ldots, y_q, x_1, \ldots, x_{n-q})$? O, dicho con otras palabras, ¿existe alguna manera de crear un nuevo programa Q que se comporte como P, asumiendo que en P los q primeros parámetros fueran obligatoriamente $y_1, \ldots, y_q$, y sólo recibiéramos los $n - q$ parámetros restantes? Si siempre es posible crear tal programa Q, explica cómo se crearía en el caso general y muestra qué aspecto tendría en el caso concreto en que $n = 5$, $q = 2$, $y_1 = 1$ y $y_2 = 2$. Si no siempre es posible crear tal programa, explica por qué.

3. Demuestra que si A y B son un conjuntos infinitos decidibles, entonces $A \cup B$ es el rango de una función recursiva creciente.

4. Responde a las siguientes cuestiones:

   (a) Supongamos que A y B son conjuntos recursivos. ¿Se cumple siempre que $A \backslash B$ es recursivo? Si es así, demuestra por qué. Si no, muestra un contraejemplo.

   (b) Supongamos que A y B son conjuntos recursivamente enumerables. ¿Se cumple siempre que $A \backslash B$ es recursivamente enumerable? Si es así, demuestra por qué. Si no, muestra un contraejemplo.

5. A continuación se muestran dos modelos de cómputo:

   (I) Los programas $whilefor$ son los definidos a través de las siguientes reglas de formación de programas:

   $$\begin{aligned} S ::= \quad & X_i := X_i + 1 \mid X_i := X_i \dot{-} 1 \mid X_i := 0 \mid \\ & \textbf{if } X_i = 0 \textbf{ then } S_1 \textbf{ else } S_2 \mid (S_1; S_2) \mid \\ & \textbf{whilefor } X_i \neq 0 \textbf{ do } S_1 \end{aligned}$$

   donde el significado de la última construcción es ejecutar S_1 mientras se cumplan las dos siguientes condiciones: (i) X_i es distinto de 0; y (ii) el número de veces que se ha ejecutado S_1 es menor o igual que el valor que tenía X_i cuando el programa alcanzó la instrucción **whilefor**.

   (II) Los programas if_sin_else son los definidos a través de las siguientes reglas de formación de programas:

   $$\begin{aligned} S ::= \quad & X_i := X_i + 1 \mid X_i := X_i \dot{-} 1 \mid X_i := 0 \mid \\ & \textbf{if } X_i = 0 \textbf{ then } S_1 \mid (S_1; S_2) \mid \\ & \textbf{while } X_i \neq 0 \textbf{ do } S_1 \end{aligned}$$

   Para cada uno de los modelos mostrados en (I) y (II), haz lo siguiente:

   (1) Si crees que la clase de funciones que se pueden computar con dicho modelo es la clase de las funciones recursivas parciales, (a) explica por qué los programas que podemos escribir en dicho modelo se pueden simular con programas *while* **o** con máquinas MRI (lo que prefieras); y (b) explica por qué los programas *while* **o** las máquinas MRI (lo que prefieras) se pueden simular con los programas de dicho modelo.

   (2) Si crees que la clase de funciones que se pueden computar con dicho modelo es la clase de las funciones recursivas primitivas, (a) explica por qué los programas que podemos escribir en dicho modelo se pueden simular con funciones recursivas primitivas **o** con programas for (lo que prefieras); y (b) explica por qué las funciones recursivas primitivas **o** los programas for (lo que prefieras) se pueden simular con los programas de dicho modelo.

   NOTA: Los programas for son iguales a los programas *while* salvo que se sustituye la instrucción **while** $X_i \neq 0$ **do** S_1 por la instrucción **for** X_i **do** S_1, la cual ejecuta S_1 un número de veces igual al valor de X_i cuando el programa alcanzó la instrucción **for**.

   (3) Si crees que el tipo de funciones que se pueden computar con dicho modelo es otro, justifícalo e identifica con la mayor precisión que puedas cómo serán dichas funciones.

6. Responde a las siguientes cuestiones:

   (a) Supongamos que φ es una función recursiva inyectiva. ¿Se cumple siempre que la función inversa de φ, es decir φ^{-1}, es recursiva? Si se cumple, demuéstralo. Si no es así, muestra un contraejemplo y justifícalo.

   (b) Supongamos que $P(x, y, z)$ es un predicado recursivo. ¿Se cumple siempre que el predicado $\exists\, y\, \exists\, z\, P(x, y, z)$ es recursivamente enumerable? Si es así, demuestra por qué. Si no, muestra un contrajemplo.

(c) Sea φ una función recursiva parcial sobreyectiva (e.d., tal que $ran(\varphi) = \mathbb{N}$). ¿Se cumple necesariamente que φ es recursiva? Si es así, demuéstralo. Si no es así, muestra un contraejemplo.

(d) Sea A un conjunto r.e. y B un conjunto recursivo tales que $A \cup B = \mathbb{N}$ y $A \cap B$ es un conjunto recursivo. ¿Se cumple necesariamente que A es recursivo? Si es así, demuéstralo. Si no, muestra un contraejemplo.

7. Sea $K_1 = \{x \mid \varphi_x(x) \simeq 7\}$. Demuestra que el conjunto K_1 es indecidible. Para ello, toma la demostración que vimos en clase para probar que el conjunto $K = \{x \mid \varphi_x(x)\downarrow\}$ es indecidible, y modifica dicha demostración para crear una demostración similar que pruebe que K_1 es indecidible.

8. Supongamos que A y B son conjuntos r.e. infinitos. Demuestra que existe una función recursiva φ tal que el conjunto de valores devueltos por φ para las entradas pares es igual a A, y el conjunto de valores devueltos por φ para las entradas impares es igual a B. Es decir:

$$\begin{aligned} A &= \{\varphi(0), \varphi(2), \varphi(4), \varphi(6), \varphi(8), \ldots\} \\ B &= \{\varphi(1), \varphi(3), \varphi(5), \varphi(7), \varphi(9), \ldots\} \end{aligned}$$

9. Las máquinas MRI' son iguales a las máquinas MRI salvo que cambia el significado de la instrucción $J(n, m, k)$, que pasa a significar lo siguiente: *si* $R_n < R_m$ *entonces salta a la instrucción* k*; en otro caso, pasa a la instrucción siguiente.* ¿Tienen las máquinas MRI' la misma capacidad de cómputo que las máquinas MRI? Si es así, demuestra que cualquier programa MRI se puede convertir en un programa MRI' equivalente. Si no es así, explica por qué.