Ejercicio 1. Definir macros en el lenguaje $\mathcal{S}++$ para las siguientes pseudo-instrucciones, donde k es una constante:

a. $V = 0$

$V := 0$: While $V \neq 0$:
    $V--$

b. $V = k$

$V := k$

$V := 0$
$\left.\begin{array}{l} V{+}{+} \\ \vdots \\ V{+}{+} \end{array}\right\} k \text{ veces}$

c. $V_1 = V_2$

$V_1 := V_2$: $V_1 := 0$
While $V_2 \neq 0$ do {
    $Z_0{+}{+}$ — ✓ fresco (empieza en 0)
    V_2-- }
While $Z_0 \neq 0$ do {
    $V_1{+}{+}$
    $V_2{+}{+}$
    Z_0-- }

d. $V_1 = V_1 + V_2$

$V_1 += V_2$

$\left.\begin{array}{l} V_1{+}{+} \\ \vdots \\ V_1{+}{+} \end{array}\right\} V_2 \text{ veces}$

$Z_1 := V_2$
While $Z_1 \neq 0$
    $V_1{+}{+}$
    Z_1--
✓

e. **If** $V \neq 0$ **then** P_1 **else** P_2

$Z_1 := V$
$Z_2 := 1$
While $Z_1 \neq 0$ {
    P_1
    Z_2--
    $Z_1 := 0$ → salir While
}
While $Z_2 \neq 0$ {
    $P2$
    $Z2--$ → salir While
}

f. **loop** (loop infinito)

While $Z_1 \neq 0$
    nop

ó

$Z_1 := 1$
While $Z_1 \neq 0$
    $Z_1{+}{+}$
    Z_1-- (si pueden dar la vuelta)

g. $V = \Psi_P^n(V_1, ..., V_n)$ → Esto es guardar el res de un programa (los Z_m de cada macro son frescos siempre)

$\left.\begin{array}{l} Z_0 := X_1 \\ \vdots \\ Z_m := X_m \end{array}\right\}$ Guardar x's actuales

$\left.\begin{array}{l} X_1 := V_1 \\ \vdots \\ X_m := V_m \end{array}\right\}$ Poner parametros

P

$V := Y$

$\left.\begin{array}{l} X_1 := Z_0 \\ \vdots \\ X_m := Z_m \end{array}\right\}$ restaurar

Ejercicio 2.

a. Dar un pseudo-programa en el lenguaje $\mathcal{S}++$ que compute la función de dos variables $f(x_1, x_2) = x_1 + x_2$. Se pueden usar las macros definidas en el ejercicio 1.

```
Y := 0
Y := X1
Y += X2
```

b. Dar el programa que resulta de expandir todas las macros utilizadas en el inciso anterior. Prestar especial atención a la instanciación de variables frescas.

ASUMO Y=0, sino incluyo → ? Lo pongo en 0 o que no lo repita

```
While X1 ≠ 0 do {
    Z0 ++
    X1 -- }
While Z0 ≠ 0 do {
    Y ++
    X1 ++
    Z0 -- }
While X2 ≠ 0 do {
    Z2 ++
    X2 -- }
While    ≠ 0 do {
    Z1 ++
    X2 ++
    Z2 -- }
While Z1 ≠ 0 do {
    Y ++
    Z1 -- }
```

Y = X1

Z1 = X2

Y = Y + X2

c. Sea P el programa obtenido en el ejercicio anterior. Determinar cuáles son las siguientes tres funciones computadas por P según la cantidad de argumentos que recibe:

$$\Psi_P^1 : \mathbb{N} \to \mathbb{N} \qquad \Psi_P^2 : \mathbb{N}^2 \to \mathbb{N} \qquad \Psi_P^3 : \mathbb{N}^3 \to \mathbb{N}$$

Ψ_P^1 Computa $f(x_1) = x_1$ ya que $x_2 = 0$ inicialmente

Ψ_P^2 Computa $f(x_1) = x_1 + x_2$

Ψ_P^3 Computa $f(x_1, x_2, x_3) = x_1 + x_2$ ya que x_3 sería 0 y no se usa.

Ejercicio 3. Sea $p : \mathbb{N}^n \to \{0, 1\}$ un predicado computable. Definir macros en el lenguaje $\mathcal{S}++$ para las siguientes pseudo-instrucciones:

a. **While** $p(x_1, ..., x_n)$ **do** P

While P(X1...Xn) do { P } :

```
P(X1,... Xn)    ← resultado en Y
Z1 := Y
Y := 0
While Z1 ≠ 0 do {
    P
    P(X1,...Xn)
    Z1 := Y
    Y := 0
```

→ si fuera, hacer ≠ 1, se puede negar

suponiendo que es un, mientras cumpla.

- V1 :

```
if V ≠ 0 then {
    V -- }
Else { V ++ }
```

b. **If** $p(x_1, ..., x_n)$ **then** P_1 **else** P_2

```
P(X1,...Xm)
Z1 := Y
Y := 0
Z2 := 1

While Z1 ≠ 0 {
    P1
    Z2 --
    Z1 --    }
While Z2 ≠ 0 {
    P2
    Z2 --}
```

Lo mismo, en if P = 0 o 1

? da igual, esto el not

Ejercicio 4. Dadas las funciones parciales computables $f, g : \mathbb{N} \to \mathbb{N}$:

$$f(x) = \begin{cases} 1 & \text{si x=3} \\ \uparrow & \text{si no} \end{cases} \qquad g(x) = 2x$$

Demostrar que la siguiente función es parcial computable:

$$h(x) = \begin{cases} f(x) & \text{si } x \geq 5 \vee x = 3 \\ g(x) & \text{si no} \end{cases}$$

Se que f y g son parciales computables, existen F y G tal que $\Psi_F^{(1)}(x) = f(x)$ y $\Psi_G^{(1)} = g(x)$

Puedo construir un programa H tal que $\underbrace{\Psi_H^{(1)}(x) = h(x)}_{H \text{ computa } h}$. Usando F y G

H:

```
if( P )then{
  F}
Else{
  g}
```

Basta ver que P, el predicado tal que computa $x \geq 5 \vee x = 3$ $\Psi_P^{(1)} = p(x)$

$$p(x) = \begin{cases} 1 & \text{si igual3}(x) \vee \text{mayor=5}(x) \\ 0 & \text{si no} \end{cases}$$

```
P: if(igual3(X1)) then{
      Y:=1
   Else
     if(mayor=5(X)) then{
        Y:=1}
     Else{
        Y:=0 }}
```

Falta ver que mis 2 nuevos predicados sean computables

```
igual 3:
    Z1:=X1
    Z2:=2   (X1-1)
 While(Z2≠0){
    Z2--
    Z1--}
 if(Z1≠0)then      //Caso Z1>2
   Z1--
   if(Z1≠0)then  // Caso Z1>3
     Y:=0
   Else          //Z1==3
     Y:=0
 Else           //Caso Z1<2
```

```
mayor=5 (x):
     Z1:=X1
     Z2:=4     (X1-1)
     While(Z2≠0){
       Z2--
       Z1-- {
     if(Z1≠0)then
        Y:=1
     Else
        Y:=0
```

→ quedo mayor a 0 luego de restar 4. Eso 5 o mas.

Con todo mostramos que H, que computa $h(x)$ es un programa valido

∴ $h(x)$ es parcial computable

Ejercicio 5.

a. Sea $r : \mathbb{N}^{n+1} \to \{0, 1\}$ un predicado computable. Mostrar que la siguiente función es parcialmente computable:

$$\mu_r(x_1, ..., x_n, y) = \begin{cases} min\{t \mid t \geq y \wedge r(x_1, ..., x_n, t)\} & \text{si existe tal } t \\ \uparrow & \text{si no} \end{cases}$$

b. Usando el resultado anterior, demostrar que si una función $f : \mathbb{N} \to \mathbb{N}$ es biyectiva y computable, su inversa f^{-1} también es computable.

a) $\mu_R(x_1, \dots x_m, y)$ devuelve el minimo t mayor o igual a y que $r(x_1, \dots x_m, t)$, si el t no existe, lo busca eternamente $(\uparrow)$

Hay que dar un programa tal que $\Psi_U^{(m+1)}(\dots) = \mu_R(\dots)$. Ademas R es computable

```
U:
  t:=y
  While(¬R){
    t++
  }
  Y:=t
```

seria While not. Asumo que toma t. Usamos X_{m+m} reservado.

Ver desde y hasta ∞, si existe $t \geq y$ da tenY. Sino $\uparrow$

b) Sabemos que F es computable. Osea hay un programa $F / \Psi_F^{(m)}(\dots) = f(x)$

y es biyectiva → si $x_1 \neq x_2 \to f(x_1) \neq f(x_2)$
$\wedge \ \forall y \in \mathbb{N} \ \exists x / f(x) = y$

Buscamos que $\exists G / \Psi_G^{(m)}(\dots) = f^{-1}(\dots)$. Como es biyectiva se que hay un unico valor en el dominio, y existe. Antes dado un unico X que de lo y, ahora hay un unico y que da X.

Usemos μ_R, con $y=0$ y buscamos $t \geq 0$ tal que $f(t)=x$. R sea $R(y,x)$

Sea $R(y,x)$ el predicado $\begin{cases} 1 & \text{si } f(x)=y \\ 0 & \text{sino} \end{cases}$ (es computable) dado x, busco y. Voy probando x's

$U_R(x,0)=G$

G computable f^{-1} $\therefore$ f^{-1} es computable

Ejercicio 6. Demostrar que si $f : \mathbb{N} \to \mathbb{N}, g : \mathbb{N}^k \to \mathbb{N}$ son dos funciones computables, entonces $f \circ g$ también es computable.

$F(g(x_1,\ldots x_n))$ quiero un programa que compute eso, $\Psi_H^{(k)} = Q(x_1,\ldots x_n)$ $h(x_1,\ldots x_n) = F(g(x_1,\ldots x_n))$.

H: $g \to$ computable

$X_1 := Y$

$F \to$ computable

Teorema

Sea P un predicado computable, entonces la función definida como:

$$(\exists x)(P(x))$$

que devuelve 1 si existe tal x y se indefine si no, es parcial computable.

Ejercicio 7. Usando las funciones computables STP^n y $SNAP^n : \mathbb{N}^{n+2} \to \mathbb{N}$ vistas en clase, demostrar que las siguientes funciones son parcialmente computables:

$$f_1(x,y) = \begin{cases} 1 & \text{si } y \in \text{Dom } \Phi_x^{(1)} \\ \uparrow & \text{en otro caso} \end{cases} \qquad f_2(x) = \begin{cases} 1 & \text{si Dom } \Phi_x^{(1)} \neq \emptyset \\ \uparrow & \text{en otro caso} \end{cases}$$

$$f_3(x,y) = \begin{cases} 1 & \text{si } y \in \text{Im } \Phi_x^{(1)} \\ \uparrow & \text{en otro caso} \end{cases} \qquad f_4(x,y) = \begin{cases} 1 & \text{si Dom } \Phi_x^{(1)} \cap \text{Dom } \Phi_y^{(1)} \neq \emptyset \\ \uparrow & \text{en otro caso} \end{cases}$$

$$f_5(x) = \begin{cases} 1 & \text{si } \Phi_x^{(1)} \text{ tiene un punto fijo} \\ \uparrow & \text{en otro caso} \end{cases} \qquad f_6(x,y) = \begin{cases} 1 & \text{si existe algún } z \text{ tal que } \Phi_x^{(1)}(z) = \Phi_y^{(1)}(z) \\ \uparrow & \text{en otro caso} \end{cases}$$

$$f_7(x,y) = \begin{cases} 1 & \text{si } \Phi_x^{(1)}(y) \downarrow \text{ y } \Phi_x^{(1)}(y) \text{ es primo} \\ \uparrow & \text{en otro caso} \end{cases} \qquad f_8(x,y) = \begin{cases} 1 & \text{si existe algún } t \leq y \text{ tal que } \Phi_x^{(1)}(t) \downarrow \\ \uparrow & \text{en otro caso} \end{cases}$$

1: Pea un programa de número e

$\Psi_P^{(m)} = \Phi_e^{(m)}$

STEP

El predicado $STEP^{(n)}(e,t,x_1,\ldots,x_n) : \mathbb{N}^{n+2} \to \mathbb{N}$ devuelve 1 si el programa de número e con entrada $x_1,\ldots,x_n$ termina su ejecución tras t o menos pasos, y 0 sino.

SNAP

Definimos la función $SNAP^{(n)} : \mathbb{N}^{n+2} \to \mathbb{N}$, tal que $SNAP(e,t,x_1,\ldots,x_n)$ codifica la configuración instantánea del programa de número e con entrada $x_1,\ldots,x_n$ en tiempo t.

1) 1 si y pertenece al dominio del numero de programa X.

Idea, puedo con y, si no se colgó, da algo 1, sino $\uparrow$, un programa que lo compute. Sea X, el programa para X. Entonces $\Psi_F^{(2)} = f_1(x,y)$

Sino: Escribo la función como (Geo)

F:
X
$Y := 1$

$F_1(x,y) = \exists t\,(STP^{(1)}(x,t,y)=1)$ Como Step es computable, F1 lo es.

2)

$$f_2(x) = \begin{cases} 1 & \text{si Dom } \Phi_x^{(1)} \neq \emptyset \\ \uparrow & \text{en otro caso} \end{cases}$$

Como uso con codificación de tuplas, entonces con 1 contador, tengo "2" (00, 10, 01, 11, 21, ... 2,0 ... 0,2 12, 22 ...) $\to$ todo $\mathbb{N}^2$

$\exists u\,(STP(x, \underset{t}{l(u)}, \underset{y\,(x_1,\text{el param})}{R(u)})=1)$ Es parcial computable, voy probando tiempo y entrada. Si no hay y válido, el dom es $\emptyset$, y se indefine buscando.

$$f_3(x,y) = \begin{cases} 1 & \text{si } y \in \text{Im } \Phi_x^{(1)} \\ \uparrow & \text{en otro caso} \end{cases}$$

quiero ver si y es algún resultado

variable

índice de Y

$F_3(x,y) = (\exists \langle t,z\rangle)(Step(x,t,z)=1 \wedge R(SNAP(x,t,z))[0]=y)$

Ver para que t termina, y veo "la foto" final

$$f_4(x,y) = \begin{cases} 1 & \text{si Dom } \Phi_x^{(1)} \cap \text{Dom } \Phi_y^{(1)} \neq \emptyset \\ \uparrow & \text{en otro caso} \end{cases}$$

$F_4(x,y) = (\exists \underbrace{\langle z, \langle t_1,t_2\rangle\rangle}_{\text{2 doble codificación}}) = (Step(x,t_1,z)=1 \wedge Step(y,t_2,z)=1)$

$$f_5(x) = \begin{cases} 1 & \text{si } \Phi_x^{(1)} \text{ tiene un punto fijo} \\ \uparrow & \text{en otro caso} \end{cases}$$

$\to$ Aparentemente que $f(x)=x$

$f(z)=z$ en este caso

$(\exists \langle t,z\rangle)(Step(x,t,z)=1 \wedge R(Snap(x,t,z))[0]=z) = f_5(x)$

$$f_6(x,y) = \begin{cases} 1 & \text{si existe algún } z \text{ tal que } \Phi_x^{(1)}(z) = \Phi_y^{(1)}(z) \\ \uparrow & \text{en otro caso} \end{cases}$$

$x(z) = y(z)$

$f_6(x,y) = (\exists \langle\langle t_1,t_2\rangle,z\rangle)(Step(x,t_1,z)=1 \wedge Step(y,t_2,z) \wedge (R(Snap(x,t,z))[0] = R(Snap(y,t_2,z))[0]))$

$$f_7(x, y) = \begin{cases} 1 & \text{si } \Phi_x^{(1)}(y)\downarrow \text{ y } \Phi_x^{(1)}(y) \text{ es primo} \\ \uparrow & \text{en otro caso} \end{cases}$$

$$f_7(x,y) = (\exists \langle t, y \rangle)(\mathrm{STEP}(x,t,y) = 1 \wedge \mathrm{Esprimo}(R(\mathrm{Snap}(x,t,y))[0]) = 1).$$

Es computable si es primo lo es, que como es un natural finito lo va a ser :)

$$f_8(x, y) = \begin{cases} 1 & \text{si existe algún } t \leq y \text{ tal que } \Phi_x^{(1)}(t)\downarrow \\ \uparrow & \text{en otro caso} \end{cases}$$

$\leq$ es un predicado computable

$$f_8(x,y) = (\exists \langle t, c \rangle)(\mathrm{Step}(x,c,t) = 1 \wedge (t \leq y))$$

contador...
no se porque necesito T para la variable

Luego de $t > y$ siempre se indefine.