Algoritmos y Estructuras de Datos II

Especificación

Departamento de Computación Facultad de Ciencias Exactas y Naturales Universidad de Buenos Aires

Trabajo Práctico 1: Especificación

Alias del grupo: WOWOANERNCTYDOHJTMEA

Integrante	LU	Correo electrónico
Barragán, Jerónimo	1472/21	barragan.jeronimo123@gmail.com
Cardozo, Alfredo	820/19	mail@alfredo.com
García Alurralde, Jorge	437/22	jalurralde@dc.uba.ar
Sarkissian, Ralph	1698/21	ralphsarkissian.b@gmail.com

Reservado para la cátedra

Instancia	Docente	Nota
Primera entrega		
Segunda entrega		

1. Renombres de tipos

```
TAD Persona es STRING
```

TAD Compra es TUPLA(Puesto, STRING, NAT)

TAD Promo es TUPLA(NAT, NAT)

2. TAD Lollapatuza

```
TAD Lollapatuza
                                      festival
         géneros
         exporta
                                      festival, generadores, observadores, estáEnElFestival, stock, hayStock?, másGastó, cuantoGastó
          usa
                                      BOOL, NAT, SECUENCIA(\alpha), CONJUNTO(\alpha), MULTICONJUNTO(\alpha), DICCIONARIO(CLAVE, SIGNIFI-
                                      CADO), STRING, PUESTO, PERSONA, COMPRA
         igualdad observacional
                                     (\forall f_1, f_2: \text{ festival}) \ (f_1 =_{\text{obs}} f_2 \iff \begin{pmatrix} \text{Puestos}(f_1) =_{\text{obs}} \text{Puestos}(f_2) \land_{\text{L}} \\ \text{Personas}(f_1) =_{\text{obs}} \text{Personas}(f_2) \land_{\text{L}} \\ (\forall \text{per}: \text{persona}) (\text{per} \in \text{Personas}(f_1) \Rightarrow_{\text{L}} \\ \text{Compras}(f_1, \text{per}) =_{\text{obs}} \text{Compras}(f_2, \text{per})) \end{pmatrix}
         observadores básicos
              Puestos : festival \longrightarrow conj(puesto)
              Personas : festival f \longrightarrow \text{conj}(\text{persona})
              Compras : festival f \times \text{persona } per \longrightarrow \text{multiconj}(\text{compra})
                                                                                                                                                                                             {estáEnElFestival?(f, per)}
          generadores
              abrir
Festival : \longrightarrow festival
              ponerPuesto : festival f \times \text{puesto } p \longrightarrow \text{festival } \{p \notin \text{Puestos}(f) \land (\forall q : \text{puesto}) \ (q \in \text{Puestos}(f) \Rightarrow_L
                                                                                                                                 (\forall \text{item} : \text{string}) \text{ (vende?(p, item)} \land \text{vende?(q, item)} \Rightarrow_{L}
                                                                                                                                      precio(p, item) =_{obs} precio(q, item)))
              agregar
Persona : festival f \times \text{persona } per \longrightarrow \text{festival}
                                                                                                                                                                                       {¬ estáEnElFestival?(f, per)}
              comprar<br/>Item : festival f \times puesto p \times persona per \times string item \times nat<br/> unidades \longrightarrow festival
                                                                                                                        \int (p \in Puestos(f) \land vende?(p, item)) \land_L
                                                                                                                       estáEnElFestival?(f, per) \(\lambda\) hayStock?(p, ítem, unidades)
              hackear : festival f \times \text{persona } p \times \text{string } item \longrightarrow \text{festival}
                                            \int est \acute{a} En El Festival? (f, per) \wedge_L (\exists \ c : compras) (c \in Compras (f, per) \wedge_L (\acute{a} tem De (c) =_{obs} \acute{a} tem \wedge_L)
                                            \cot Gast \circ EnCompra(f, per, c) = cost = cost
         otras operaciones
              está
En<br/>El
Festival? : festival f \times \text{persona } per \longrightarrow \text{bool}
                                                                                                                                                                          \{p \in Puestos(f) \land vende?(p, item)\}
              stock : festival f \times \text{puesto } p \times \text{string } item \longrightarrow \text{nat}
              hayStock? : festival f \times \text{puesto } p \times \text{string } item \times \text{nat } unidades \longrightarrow \text{bool}
                                                                                                                                                                       \{p \in Puestos(f) \land vende?(p, item)\}
              másGastó : festival f \longrightarrow persona
                                                                                                                                                                                                    \{\neg vacio?(Personas(f))\}
              más
Gastó<br/>Aux : festival f \times \text{conj}(\text{persona}) \ c \longrightarrow \text{persona}
                                                                                             \{\neg vac\'io?(c) \land_L (\forall per: persona) \ (per \in c \Rightarrow_L est\'aEnElFestival?(f, per))\}
              cuánto
Gastó : festival f \times \operatorname{persona} per \longrightarrow \operatorname{nat}
                                                                                                                                                                                            {estáEnElFestival?(f, per)}
              cuánto
Gastó
Aux : festival f \times \text{persona } per \times \text{multiconj}(\text{compra}) \longrightarrow \text{nat}
                                                                                                                                                                                             {estáEnElFestival?(f, per)}
              cuánto
Gastó
En<br/>Compra : festival f \times persona per \times compra c \longrightarrow nat
                                                                                                                                                                                             {estáEnElFestival?(f, per)}
              compra<br/>AHackear : festival f \times persona per \times multiconj<br/>(compra) compras \times string item \longrightarrow compra
                                              \label{eq:cuantoGastoEnCompra} \begin{cases} \text{cuántoGastoEnCompra}(f,\,\text{per},\,c) = \text{precio}(\text{puestoDe}(c),\,\text{ítem}) \times \text{cantidadDe}(c))) \end{cases}
              {\tt puestoDe} \; : \; \widehat{\tt compra} \; \longrightarrow \; puesto
              itemDe : compra \longrightarrow string
              cantidadDe \ : \ compra \ \ \longrightarrow \ nat
                                      \forall f: festival, \forall p, p_1, p_2: puesto, \forall per, per_1, per_2: persona, \forall item, item_1, item_2: string,
          axiomas
                                      \forall unidades: nat, \forall compras: multiconj(compra), \forall compra: compra
              Puestos(abrirFestival) \equiv \emptyset
              Puestos(ponerPuesto(f, p)) \equiv Ag(p, Puestos(f))
              Puestos(agregarPersona(f, per)) \equiv Puestos(f)
              Puestos(comprarÍtem(f, p, per, item, unidades)) \equiv Puestos(f)
              Personas(abrirFestival) \equiv \emptyset
```

 $Personas(ponerPuesto(f, p)) \equiv Personas(f)$

```
Personas(agregarPersona(f, per)) \equiv Ag(per, Personas(f))
Personas(comprarItem(f, p, per_2, item, unidades), per_1) \equiv Personas(f)
Compras(ponerPuesto(f, p), per) \equiv Compras(f, per)
\text{Compras}(\text{agregarPersona}(\mathbf{f}, \, per_2), \, per_1) \, \equiv \, \mathbf{if} \, \, per_1 =_{\text{obs}} \, per_2 \, \, \mathbf{then} \, \, \emptyset \, \, \mathbf{else} \, \, \text{Compras}(\mathbf{f}, \, per_1) \, \, \mathbf{fi} \, \,
Compras(comprarItem(f, p, per_2, item, unidades), per_1) \equiv if per_1 =_{obs} per_2
                                                                         then Ag(\langle p, \text{ item, unidades} \rangle, Compras (f, per_1))
                                                                         else Compras(f, per_1)
estáEnElFestival?(f, per) \equiv per \in Personas(f)
stock(ponerPuesto(f, p_2), p_1, item) \equiv if p_1 = obs p_2 then stockDe(p_1, item) else stock(f, p_1, item) fi
stock(agregarPersona(f, per), p, item) \equiv stock(f, p, item)
\operatorname{stock}(\operatorname{comprar}\operatorname{Item}(f, p, \operatorname{per}, item_2, \operatorname{unidades}), p, item_1) \equiv \operatorname{if} item_1 =_{\operatorname{obs}} item_2
                                                                         then stock(f, p, item_1) - unidades
                                                                         else stock(f, p, item_1)
hayStock?(f, p, item, unidades) \equiv unidades \leq stock(f, p, item)
m\acute{a}sGast\acute{o}(f) \equiv m\acute{a}sGast\acute{o}Aux(f, Personas(f))
másGastóAux(f, personas) \equiv if \#(personas) =_{obs} 1 then dameUno(personas)
                                    else if cuántoGastó(f, dameUno(personas)) \ge
                                            cuántoGastó(dameUno(sinUno(personas)))
                                            then másGastó(f, Ag(dameUno(personas), sinUno(sinUno(personas)))
                                            else másGastó(f, sinUno(personas)
                                    fi fi
cuántoGastó(f, per) \equiv cuántoGastóAux(f, per, Compras(f, per))
cuántoGastóAux(f, per, compras) \equiv if compras =_{obs} \emptyset
                                               then 0
                                               else cuántoGastóEnCompra(f, per, dameUno(compras)) +
                                                     cuántoGastóAux(f, per, sinUno(compras))
cuántoGastóEnCompra(f, per, compra) = if hayPromo?(puestoDe(compra), ítemDe(compra), cantidadDe(compra))
                                                     then aplicar Descuento(cantidad De(compra) \times precio(puesto De(compra),
                                                            descuento(puestoDe(compra), ítemDe(compra),
                                                            cantidadDe(compra)))
                                                     else cantidadDe(compra) \times precio(puestoDe(compra), item<math>De(compra))
Puestos(hackear(f, per, item)) \equiv Puestos(f)
estáEnElFestival?(hackear(f, per_2, ítem), per_1) \equiv estáEnElFestival?(f, per_1)
Compras(hackear(f, per_2, item), per_1) \equiv if per_2 \neq_{obs} per_1 then Compras(f, per_1) else Compras(f, per_1) -
                                                 \{\text{compraAHackear}(f, per_1, \text{Compras}(f, per_1, \text{item}), \text{item})\}\ \mathbf{fi}
stock(hackear(f, per, item_2), p, item_1) \equiv if item_1 = obs item_2 \land puestoDe(compraAHackear(f, per, item_1)) = obs p
                                                    then stock(f, p, item_1) + cantidadDe(compraAHackear(f, per, item_1))
                                                    else stock(f, p, ítem_1)
compraAHackear(f, per, compras, item) = if esHackeable(f, per, dameUno(compras), item) then dameUno(compras)
                                                   else compraAHackear(per, sinUno(compras), ítem) fi
esHackeable(f, per, compra, ítem) \equiv ítemDe(compra) =_{obs} ítem \land_L cuántoGastóEnCompra(f, per, compra) =_{obs}
                                            precio(puestoDe(compra), ítem) × cantidadDe(compra)
puestoDe(compra) \equiv \pi_1(compra)
ítemDe(compra) \equiv \pi_2(compra)
cantidadDe(compra) \equiv \pi_3(compra)
```

Fin TAD

3. TAD Puesto

```
TAD Puesto
      géneros
                         puesto
      exporta
                         puesto, generadores, observadores, vende?, precio, hayPromo?, descuento, stockDe
                         Bool, Nat, Secuencia(\alpha), Diccionario(clave, Significado), String, Promo
      igualdad observacional
                         (\forall p_1, p_2: puesto) (p_1 =_{obs} p_2 \iff \begin{pmatrix} \operatorname{Men\acute{u}}(p_1) =_{obs} \operatorname{Men\acute{u}}(p_2) \wedge \operatorname{Stock}(p_1) =_{obs} \operatorname{Stock}(p_2) \\ \wedge \operatorname{Promos}(p_1) =_{obs} \operatorname{Promos}(p_2) \end{pmatrix}
      observadores básicos
         Men\acute{u} : puesto \longrightarrow dicc(string, nat)
         Stock : puesto \longrightarrow dicc(string, nat)
         Promos: puesto \longrightarrow dicc(string, secu(promo))
      generadores
         abrirPuesto: \longrightarrow puesto
                                                                                                                                \{\neg def?(item, Stock(p))\}
         agregar
Ítem : puesto p \times \text{string } \textit{item} \times \text{nat } \textit{precio} \times \text{nat } \textit{stock} \longrightarrow \text{puesto}
         agregar
Promo : puesto p \times \text{string } \text{i} tem \times \text{nat } cantidad \times \text{nat } descuento \longrightarrow \text{puesto}
                                    \int descuento > 0 \land (\forall c : string)((c \in claves(Promos(p)) \land c =_{obs} item) \Rightarrow_L
                                          (\forall \ prom: promo)(est\'a?(prom, \ obtener(c, \ Promos(p))) \Rightarrow_{\scriptscriptstyle L} \pi_1(prom) \neq_{obs} cantidad))))
      otras operaciones
         vende? : puesto p \times \text{string } item \longrightarrow \text{bool}
         precio: puesto p \times \text{string } item \longrightarrow \text{nat}
                                                                                                                                          {vende?(p, item)}
                                                                                                                                          {vende?(p, item)}
         hayPromo?: puesto p \times \text{string } item \times \text{nat } cantidad \longrightarrow \text{bool}
         descuento : puesto p \times \text{string } item \times \text{nat } cantidad \longrightarrow \text{nat}
                                                                                         {vende? (p, \text{ item}) \land_L \text{ hayPromo?}(p, \text{ item}, \text{ cantidad})}
         stock
De : puesto p \times \text{string } item \longrightarrow \text{nat}
                                                                                                                                         {vende?(p, ítem)}
         {\rm cantM\acute{n}DePrim}\ :\ {\rm secu(promo)}\ s\ \longrightarrow\ {\rm nat}
                                                                                                                                                \{\neg \text{ vac\'ia?(s)}\}
                                                                                                                                          \{\neg \text{ vac}(\text{fin}(s))\}
         \operatorname{cantMinDeSeg} : \operatorname{secu}(\operatorname{promo}) s \longrightarrow \operatorname{nat}
         hayPromoAux: puesto p \times \text{string } item \times \text{nat } cantidad \times \text{secu(promo)} s \longrightarrow \text{bool}
                                                                                                              \{def?(item, Promos(p)) \land \neg vacía?(s)\}
         descuento
Aux : puesto p \times \text{string } item \times \text{nat } cantidad \times \text{secu(promo) } s \longrightarrow \text{promo}
                                                                              \{\text{hayPromoAux}(p, \text{ item}, \text{ cantidad}, \text{Promos}(p)) \land \neg \text{vac}(a?(s))\}
                         \forall p: puesto, \forall item: string, \forall precio, stock, cantidad, descuento: nat, \forall promos: secu(promo)
      axiomas
         Menú(abrirPuesto) ≡ vacío
         Menú(agregarÍtem(p, ítem, precio, stock)) = definir(ítem, precio, Menú(p))
         Menú(agregarPromo(p, ítem, cantidad, descuento)) 

Menú(p)
         Stock(abrirPuesto) \equiv vacío
         Stock(agregarItem(p, item, precio, stock)) \equiv definir(item, stock, Stock(p))
         Stock(agregarPromo(p, item, cantidad, descuento)) \equiv Stock(p)
         Promos(abrirPuesto) ≡ vacío
         Promos(agregarItem(p, item, precio, stock)) \equiv Promos(p)
         Promos(agregarPromo(p, ítem, cantidad, descuento)) = if def?(ítem, Promos(p))
                                                                                             then definir(ítem, <cantidad, descuento> •
                                                                                                      obtener(ítem, Promos(p)), Promos(p))
                                                                                             else definir(ítem, <cantidad, descuento> \bullet <> ,
                                                                                                    Promos(p)
                                                                                          fi
         vende?(p, item) \equiv def?(item, Menú(p))
         precio(p, item) \equiv obtener(item, Menú(p))
         hayPromo?(p, item, cantidad) \equiv if def?(item, Promos(p))
                                                            then hayPromoAux(p, ítem, cantidad, obtener(ítem, Promos(p)))
         \operatorname{cantMinDePrim}(\operatorname{promos}) \equiv \pi_1(\operatorname{prim}(\operatorname{promos}))
         \operatorname{cantMinDeSeg(promos)} \equiv \pi_1(\operatorname{prim}(\operatorname{fin}(\operatorname{promos})))
```

```
//Cabe observar que la secuencia promos nunca es vacía
      hayPromoAux(p, ítem, cantidad, promos) ≡
           if long(promos) =_{obs} 1
             then cantidad ≥ cantMínDePrim(promos)
             else if cantMínDePrim(promos) > cantidad
                    then hayPromoAux(p, item, cantidad, fin(promos))
                     else if cantMinDeSeg(promos) \le cantidad
                            then if cantMinDeSeg(promos) > cantMinDePrim(promos)
                                    then hayPromoAux(p, ítem, cantidad, fin(promos))
                                    else hayPromoAux(p, ítem, cantidad, prim(promos) • fin(fin(promos)))
                            else hayPromoAux(p, ítem, cantidad, prim(promos) • fin(fin(promos)))
                         fi
                   fi
      descuento(p, ítem, cantidad) \equiv \pi_2(\text{promoAux}(p, \text{ítem, cantidad, obtener}(\text{ítem, Promos}(p))))
      descuentoAux(p, item, cantidad, promos) \equiv
            if long(promos) =_{obs} 1
              then prim(promos)
              else if cantMinDePrim(promos) > cantidad
                      then descuentoAux(p, ítem, cantidad, fin(promos))
                      else if cantMinDeSeg(promos) \le cantidad
                               then if cantMinDeSeg(promos) > cantMinDePrim(promos)
                                       then descuentoAux(p, item, cantidad, fin(promos))
                                        else descuentoAux(p, ítem, cantidad, prim(promos) • fin(fin(promos)))
                                     fi
                               else descuentoAux(p, item, cantidad, prim(promos) \bullet fin(fin(promos)))
                    fi
      stockDe(p, item) \equiv obtener(item, Stock(p))
Fin TAD
```