/\*Resolución parcial: Repartidor -> Código | Cliente -> Paquete

Ejercicio 1: Desarrollar la función "cantidadClientesARecibirPaquete"

que describa la cantidad de clientes

que tienen su pedido en el mapa

y todavía no fue entregado

\*/

function cantidadClientesARecibirPaquete(){

/\* Propósito: Describe la cantidad de clientes que deben recibir su paquete, estos deben estar en el mapa.

Precondiciones: Ninguna

Tipo: Número \*/

IrAPrimeraCeldaEnUnRecorridoAl\_Y\_(Este, Norte)

cantidadPaquetes := unoSi\_ceroSiNo(hayClienteAcaConPaqueteSinEntregarEnElMapa())

while(haySiguienteCeldaEnUnRecorridoAl\_Y\_(Este, Norte)){

IrASiguienteCeldaEnUnRecorridoAl\_Y\_(Este, Norte)

cantidadPaquetes := cantidadPaquetes + unoSi\_CeroSiNo(hayClienteAcaConPaqueteSinEntregarEnElMapa())

}

return(cantidadPaquetes)

}

function hayClienteAcaConPaqueteSinEntregarEnElMapa(){

/\* Propósito: Indica si hay un cliente en la ubicación actual que tiene su paquete en el mapa sin entregar.

Precondiciones: Ninguna

Tipo: Boolean

\*/

return(hayClienteAca() && not tienePaqueteEntregado() && estaElPaquete\_EnElMapa(idClienteAca()) )

}

function tienePaqueteEntregado(){

/\*Proposito: Indica si el cliente en la ubicación actual tiene su pedido entregado

Prec.: \*Debe haber un cliente en la ubicación

Tipo: Boolean

\*/

return(hayPaqueteConId\_(idClienteAca()) )

}

function hayPaqueteConId\_(idPaquete){

return(hayPaqueteAca() && idPaqueteAca() == idPaquete)

}

function estaElPaquete\_EnElMapa(idPaquete){

IrAPrimeraCeldaEnUnRecorridoAl\_Y\_(Este, Norte)

while(haySiguienteCeldaEnUnRecorridoAl\_Y\_(Este, Norte) && not hayPaqueteConId\_(idPaquete())){

IrASiguienteCeldaEnUnRecorridoAl\_Y\_(Este, Norte)

}

return(hayPaqueteConId\_(idPaquete))

}

function hayClienteAca(){

/\* Propósito: Indica si hay un cliente en la celda actual

Precondiciones: Ninguna

Tipo: Boolean

\*/

return(hayBolitas(cliente()))

}

function idClienteAca(){

/\* Propósito: Indica el id del cliente en la celda actual

Precondiciones: Ninguna

Tipo: Boolean

\*/

return(nroBolitas(cliente()))

}

function hayPaqueteAca(){

/\* Propósito: Indica si hay un paquete en la celda actual

Precondiciones: Ninguna

Tipo: Boolean

\*/

return(hayBolitas(paquete()))

}

function idPaqueteAca(){

/\* Propósito: Indica el id del paquete en la celda actual

Precondiciones: Ninguna

Tipo: Boolean

\*/

return(nroBolitas(paquete()))

}

/\*

Ejercicio 2: Desarrollar "mayorIdClienteSinPaquete" que describe

el mayor ID de un cliente que no recibió el paquete,

el cual se encuentra en el mapa

\*/

function mayorIdClienteSinPaquete(){

IrAPrimeraCeldaEnUnRecorridoAl\_Y\_(Este, Norte)

mayorIdDeCliente := maximoEntre\_Y\_(0, idClienteSinPaqueteCeroSiNo())

while(haySiguienteCeldaEnUnRecorridoAl\_Y\_(Este, Norte)){

IrASiguienteCeldaEnUnRecorridoAl\_Y\_(Este, Norte)

mayorIdDeCliente := maximoEntre\_Y\_(mayorIdDeCliente, idClienteSinPaqueteCeroSiNo())

}

return(mayorIdDeCliente)

}

function idDeClienteSinPaqueteCeroSiNo(){

return(

choose

idClienteAca when (hayClienteAcaConPaqueteSinEntregarEnElMapa())

0 otherwise

)

}

/\*

Ejercicio 3: Desarrollar el procedimiento EntregarPaqueteMayor,

que asumiendo que el cabezal se encuentra sobre el repartidor

que debe llevar ese paquete, lleva el paquete al cliente.

Ayuda: En la ubicación del cliente debe quedar junto a éste; el paquete y el repartidor.

Ojo que el repartidor puede tener que llevar varios paquetes y sólo nos importa el mayor.

\*/

procedure EntregarPaqueteMayor(){

/\*Prop:Entregar el paquete al cliente con mayor id

Prec: \*El paquete del cliente con mayor Id no fue entregado y existe en el mapa

\*El cabezal se encuentra sobre el repartidor que debe entregar el paquete con mayor Id

\*/

LlevarARepartidorAPaqueteConMayorIdCliente()

LlevarPaqueteAClienteDeMayorId()

}

procedure LlevarARepartidorAPaqueteConMayorIdCliente(){

/\*Prop: Llevar al repartidor de la ubicación actual al paquete de mayor id.

Prec: \*El paquete de mayor id no fue entregado y existe en el mapa

\*El cabezal se encuentra sobre el repartidor que debe entregar el paquete con mayor id

\*/

idRepartidor := idRepartidorAca()

SacarRepartidor() // Sacar\_DeColor\_(idRepartidorAca(), repartidor())

IrAPrimeraCeldaEnUnRecorridoAl\_Y\_(Este, Norte)

while (not esPaqueteMayorConIdRepartidor(idRepartidor)){

IrASiguienteCeldaEnUnRecorridoAl\_Y\_(Este, Norte)

}

PonerRepartidor\_(idRepartidor) // Poner\_DeColor\_(idRepartidor, repartidor())

}

function esPaqueteMayorConIdRepartidor(idCodigo){

/\*Prop: Indica si hay un cógido con id \*\*idRepartidor\*\* en la celda actual

Prec: \*Debe existir en el mapa un paquete con id \*\*idRepartidor\*\*

Tipo: Booleano

\*/

return(hayPaqueteAca() && mayorIdDeClienteSinPaquete() == idCodigo)

}

procedure LlevarPaqueteAClienteDeMayorId(){

idRepartidor := idRepartidorAca()

idPaquete := idPaqueteAca()

SacarRepartidor() // Sacar\_DeColor\_(idRepartidorAca(), repartidor())

SacarPaquete() // Sacar\_DeColor\_(idRepartidorAca(), repartidor())

SacarCodigo() // Sacar\_DeColor\_(idRepartidorAca(), repartidor())

IrAPrimeraCeldaEnUnRecorridoAl\_Y\_(Este, Norte)

while (not esClienteId\_(idPaquete()){

IrASiguienteCeldaEnUnRecorridoAl\_Y\_(Este, Norte)

}

EntregarPaquete\_DelRepartidor\_(idPaquete, idRepartidor)

}

procedure EntregarPaquete\_DelRepartidor\_(idPaquete, idRepartidor){

/\*Proposito: Entregar el paquete \*idPaquete\* del repartidor \*idRepartidor\* en la ubicación actual

Prec: \* En la ubicación actual no debe haber un paquete

\* En la ubicación actual no debe haber un repartidor

Param: \*idPaquete: Número - El id del paquete a entregar

\*idRepartidor: Número - El id del repartidor que entrega el paquete

\*/

PonerPaquete\_(idPaquete)

PonerRepartidor\_(idRepartidor)

}