```
//TEMA 2

//ejercicio 1

function cantidadRepartidoresAEntregarPaquete(){

  /*

  PROPÓSITO: Describe la cantidad de repartidores que deben entregar sus paquete, estos paquetes deben estar en el mapa.

  PRECONDICIONES:

    *Ninguna

  TIPO: Número

  */

  IrAPrimeraCeldaEnUnRecorridoAl_Y_(Este, Norte)

  cantidadRepartidores := unoSi_ceroSino(hayRepartidorConPaqueteParaEntregarEnMapa())

  while(haySiguienteCeldaEnUnRecorridoAl_Y_(Este, Norte)){

    IrASiguienteCeldaEnUnRecorridoAl_Y_(Este, Norte)

    cantidadRepartidores := cantidadRepartidores + unoSi_ceroSino(hayRepartidorConPaqueteParaEntregarEnMapa())

  }

  return(cantidadRepartidores)

}


function hayRepartidorConPaqueteParaEntregarEnMapa(){

  /*

  PROPÓSITO: Describe si hay un repartidor en la ubicación actual que tiene paquete en el mapa sin entregar.

  PRECONDICIONES:

    *Ninguna

  TIPO: Booleano

  */

  return(hayRepartidorAcá() && hayPaqueteDel_EnElMapa(idRepartidorAcá()))

}
```

```
function hayPaqueteDel_EnElMapa(idRepartidor){
  /*
  PROPÓSITO: Indica si hay al menos un paquete con código **idRepartidor** en el mapa.
  PARÁMETROS:
    *idRepartidor: Número- el id del código del paquete a buscar
  PRECONDICIONES:
    *Ninguna
  TIPO: Booleano
  */
  IrAPrimeraCeldaEnUnRecorridoAl_Y_(Este, Norte)
  while(haySiguienteCeldaEnUnRecorridoAl_Y_(Este, Norte) && not hayPaqueteConIdCódigo_(idRepartidor)){
    IrASiguienteCeldaEnUnRecorridoAl_Y_(Este, Norte)
  }
  return (hayPaqueteConIdCódigo_(idRepartidor))
}

function hayPaqueteConIdCódigo_(idCodigo){
    /*
  PROPÓSITO: Inidica si en la ubicación aactual hay un paquete con id de código **idCodigo**.
  PARÁMETROS:
    *idCodigo: Número- el id del paquete que se busca el código
  PRECONDICIONES:
    *Ninguna
  TIPO: Booleano
  */
  return(hayPaqueteAcá() && hayCódigoAcá() && idCódigoAcá() == idCodigo)
}

//ejercicio 2
```

```
function mayorCantidadPaquetesSinEntregarDeUnRepartidor(){
  /*
  PROPÓSITO: Describe la mayor cantidad de paquete sin entregar que correspondan a un repartidor.
  PRECONDICIONES:
    *Debe haber una única cantidad máxima de paquetes por entregar de un repartidor.
  TIPO: Número
  */
  IrAPrimeraCeldaEnUnRecorridoAl_Y_(Este, Norte)
  cantidadPaquetes := máximoEntre_Y_(0, cantidadPaquetesPorEntregarDelRepartidorCeroSino())
  while(haySiguienteCeldaEnUnRecorridoAl_Y_(Este, Norte)){
    IrASiguienteCeldaEnUnRecorridoAl_Y_(Este, Norte)
    cantidadPaquetes := máximoEntre_Y_(cantidadPaquetes, cantidadPaquetesPorEntregarDelRepartidorCeroSino())
  }
  return(cantidadPaquetes)
}

function cantidadPaquetesPorEntregarDelRepartidorCeroSino(){
  /*
  PROPÓSITO: Describe la cantidad de paquetes por entregar del repartidor de la ubicación actual cuando hay, cero cuando no hay.
  PRECONDICIONES:
    *ninguna.
  TIPO: Número
  */
  return(choose paquetesPorEntregar_(idRepartidorAcá()) when (hayRepartidorAcá())
                0                otherwise
    )
}
```

```
function paquetesPorEntregar_(idRepartidor){
  /*
  PROPÓSITO: Describe la cantidad de paquetes por entregar del repartidor
**idRepartidor**
  PARÁMETROS:
    *idRepartidor: Número - el id del repartidor a contar los paquetes que faltan entregar
  PRECONDICIONES:
    *ninguna.
  TIPO: Número
  */
  IrAPrimeraCeldaEnUnRecorridoAl_Y_(Este, Norte)
  cantidadPaquetes := unoSi_ceroSino(hayPaqueteConIdCódigo_(idRepartidor))
  while(haySiguienteCeldaEnUnRecorridoAl_Y_(Este, Norte)){
    IrASiguienteCeldaEnUnRecorridoAl_Y_(Este, Norte)
    cantidadPaquetes := cantidadPaquetes +
unoSi_ceroSino(hayPaqueteConIdCódigo_(idRepartidor))
  }
  return (cantidadPaquetes)
}

//ejercicio 3
procedure EntregarPaquete_(idPaquete){
  /*
  PROPÓSITO: Entregar el paquete con id **idPaquete** al cliente que lo espera
  PARÁMETROS:
    *idPaquete: Número - El id del paquete a ser entregado
  PRECONDICIONES:
    *El cabezal se encuentra sobre el repartidor que debe entregar el paquete con id
**idPaquete**
    *El paquete con id **idPaquete** se encuentra en el mapa
    *El cliente que espera el paquete se encuentra en el mapa
  */
```

```
  LlevarRepartidorAPaqueteConId_(idPaquete)
  LlevarPaqueteAlCliente()
}

procedure LlevarRepartidorAPaqueteConId_(idPaquete){
  /*
  PROPÓSITO: Llevar al repartidor de la ubicación actual al paquete con id **idPaquete.
  PRECONDICIONES:
    *El paquete con id **idPaquete existe en el mapa y el mismo no fue entragado.
    *El cabezal se encuentra sobre el repartidor que debe entregar el paquete.
  */
  idRepartidor := idRepartidorAcá()
  SacarRepartidor() //Sacar_DeColor_(idRepartidorAca(), repartidor())
  IrAlPaquete_(idPaquete)
  PonerRepartidor_(idRepartidor)//Poner_DeColor_(idRepartidor, repartidor())
}

procedure IrAlPaquete_(idPaquete){
  /*
  PROPÓSITO: lleva el cabezal al paquete de id **idPaquete**
  PARÁMETROS:
    *idPaquete : Número - el id del paquete a llevar el cabezal
  PRECONDICIONES:
    *Debe estar en el mapa la ubicación del paquete de id **idPaquete**
  */
  IrAPrimeraCeldaEnUnRecorridoAl_Y_(Este, Norte)
  while(not esPaqueteDeId_(idPaquete)){
    IrASiguienteCeldaEnUnRecorridoAl_Y_(Este, Norte)
  }
}
```

```
function esPaqueteDeId_(idPaquete){
    /*
  PROPÓSITO: Indica si hay un paquete con id **idPaquete** en la celda actual.
  PRECONDICIONES:
    *Ninguna.
  TIPO: Booleano
  */
  return(hayPaqueteAcá() &&  idPaqueteAcá() == idPaquete)
}

procedure LlevarPaqueteAlCliente(){
  /*
  PROPÓSITO: Lleva el paquete de la ubicación actual al cliente que lo espera.
  PRECONDIDICIONES:
    *El cliente se encuentra en el mapa y todavía no ha recibido el paquete.
    *El repartidor que debe entregar el paquete se encuentra en la ubicación actual.
    *El paquete a entregar está en la ubicación actual.
  */
  idRepartidor := idRepartidorAcá()
  idPaquete := idPaqueteAcá()
  SacarRepartidor()
  SacarPaquete()
  SacarCódigo()
  IrAPrimeraCeldaEnUnRecorridoAl_Y_(Este, Norte)
  while(not esClienteId_(idPaquete)){
    IrASiguienteCeldaEnUnRecorridoAl_Y_(Este, Norte)
  }
  EntregarPaquete_DelRepartidor_(idPaquete,idRepartidor)
}

procedure EntregarPaquete_DelRepartidor_(idPaquete, idRepartidor){
```

```
  /*
  PROPÓSITO: Entregar el paquete *idPaquete* del repartidor *idRepartidor* en la
ubicación actual
  PARÁMETROS:
    *idPaquete: Número - el id del paquete a entregar
    *idRepartidor: Número - el id del repartidor que entrega el paquete
  PRECONDICIONES:
    *En la ubicación no debe haber un paquete
    *En la ubicación no debe haber un repartidor
  */
  PonerPaquete_(idPaquete)
  PonerRepartidor_(idRepartidor)
}
```