Pontificia Universidad Javeriana

Departamento de Ingeniería de sistemas

Estructuras de datos

Primera Entrega Proyecto de curso 2018 – 10

Integrantes:

* Karol Valentina Yate Vega
* David Leonardo Gómez Macareno
* Ericka Julieth Torres Páez

Empresa de mensajería y distribución de paquetes:

* **main():**

Análisis

Objetos conocidos:

Comandos

Objetos desconocidos:

Ninguno

Condiciones:

El comando ingresado debe existir entre las opciones.

Diseño:

Entradas:

Comandos de tipo cadena de caracteres.

Salidas:

Ejecución de los comandos.

Condiciones:

Los comandos deben existir.

* **list<Persona>loadPersons(char\* archivo, list<Persona> personas):**

Análisis

Objetos conocidos:

Archivo con datos de personas

Objetos desconocidos:

Las personas

Condiciones:

El archivo debe tener todos los datos (No se aceptan datos nulos)

Diseño:

Entradas:

Nombre del archivo a abrir de tipo char\*.

Personas que tiene el sistema de tipo lista.

Salidas:

Lista de personas actualizada..

Condiciones:

El archivo debe existir y tener un formato específico, no puede tener campos vacíos.

* **list<Paquete> loadPackages(char\* archivo, list<Persona> personas, list<Oficina> oficinas):**

Análisis

Objetos conocidos:

Archivo con los datos de los paquetes

Objetos desconocidos:

Los paquetes

Condiciones:

El archivo debe tener los datos completos (No se aceptan datos nulos)

Si una oficina ó región no existe, se agrega.

Diseño:

Entradas:

Nombre del archivo de tipo char\*

Personas registradas en el sistema, de tipo lista.

Oficinas registradas en el sistema, de tipo lisa.

Salidas:

Lista de una oficina específica, en una región específica, actualizada.

Condiciones:

El archivo debe existir y tener un formato específico, no puede tener campos vacíos.

* **list<Persona> regPersons(list<Persona> personas):**

Análisis

Objetos conocidos:

Nombre, apellido, ciudad, teléfono, id y dirección.

Objetos desconocidos:

La persona.

Condiciones:

Ninguna.

Diseño:

Entradas:

Lista de todas las personas que tiene el sistema, lista tipo Persona.

Salidas:

Lista actualizada de las personas registradas en el sistema.

Condiciones:

Ninguna.

* **list<Paquete> regPackages(list<Oficina> oficinas, list<Persona> personas)**

Análisis

Objetos conocidos:

Oficina, región y datos del paquete a agregar.

Objetos desconocidos:

Paquetes nuevos

Condiciones:

Todos los campos deben estar llenos.

Si una oficina ó región no existe, se agrega.

Diseño:

Entradas:

Oficinas registradas en el sistemas, lista tipo Oficina.

Personas registradas en el sistema, lista tipo Persona.

Salidas:

Lista de oficinas y cada una de sus regiones, actualizadas.

Condiciones:

El identificador del paquete no debe existir en la lista en el que será agregado.

El identificador debe contener números y letras.

* void countPackages():

Análisis

Objetos conocidos:

Paquetes, sus oficinas y regiones

Objetos desconocidos:

Ninguno

Condiciones:

Debe haber paquetes, oficinas y regiones

Diseño:

Entradas:

Los datos del sistema

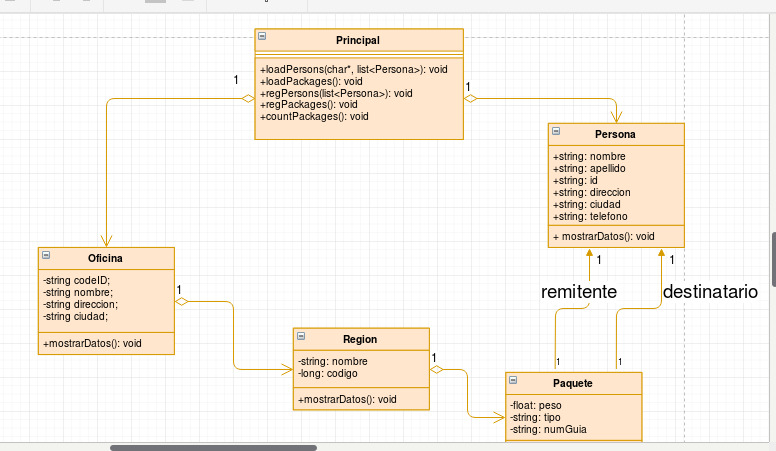
Salidas:

Impresiones en pantalla con los datos de los paquetes en cada región

Condiciones:

Debe haber paquetes en el sistema

Diagrama de Clases:



Descripción de los TADs:

Principal

• oficinas, lista de oficinas.

• personas, lista de personas.

OficinaReparto:

• codeID, cadena de caracteres alfanuméricos, 3 letras + 5 dígitos.

• nombre, cadena de caracteres con el nombre de la oficina.

• direccion, cadena de caracteres con una direccion.

• ciudad, cadena de caracteres con el nombre de una ciudad.

• regiones, lista de regiones.

Region:

• nombre, cadena de caracteres con el nombre de la región.

• codigo, cadena de caracteres alfanuméricos.

• paquetes, lista de Paquetes

• oficina, apuntador de tipo oficina a su oficina

Persona:

• nombre

• apellido, cadena de caracteres.

• identificacion, cadena de caracteres alfanuméricos.

• direccion, cadena de caracteres que forman una dirección.

• ciudad, cadena de caracteres con el nombre de una ciudad.

• telefono, cadena de caracteres numericos.

Paquete:

• remitente, objeto de tipo Persona.

• destinatario, objeto de tipo Persona diferente al remitente.

• peso, tipo float, se da en gramos.

• tipo de contenido, cadena de caracteres.

• numGuia, cadena de caracteres alfanuméricos.

• region, apuntador de tipo region a la región donde será entregado