Auxiliatura INF-131 "C"
Estructura de Datos y Algoritmos

Univ. Miguel Angel Quispe Mamani
Universidad Mayor de San Andrés
Carrera de Informática

20/03/2023

Problema

- **Pila de Categoría Poblacional:** Almacena → idCategoria, nombre
- **Cola de Departamentos:** Almacena → nombre de departamento

Según el orden de los Departamentos, la Múltiple Cola almacena sus Provincias

- **Múltiple Pila de Provincias:** Almacena → nombre de provincia y cada provincia tiene una **Cola de Municipios**
- **Cola de Municipios:** Almacena → nombre de municipio, idCategoría, población, residuoSólido(kg/Hab_dia), coberturaRec, disposiciónFinal, aprovechamiento

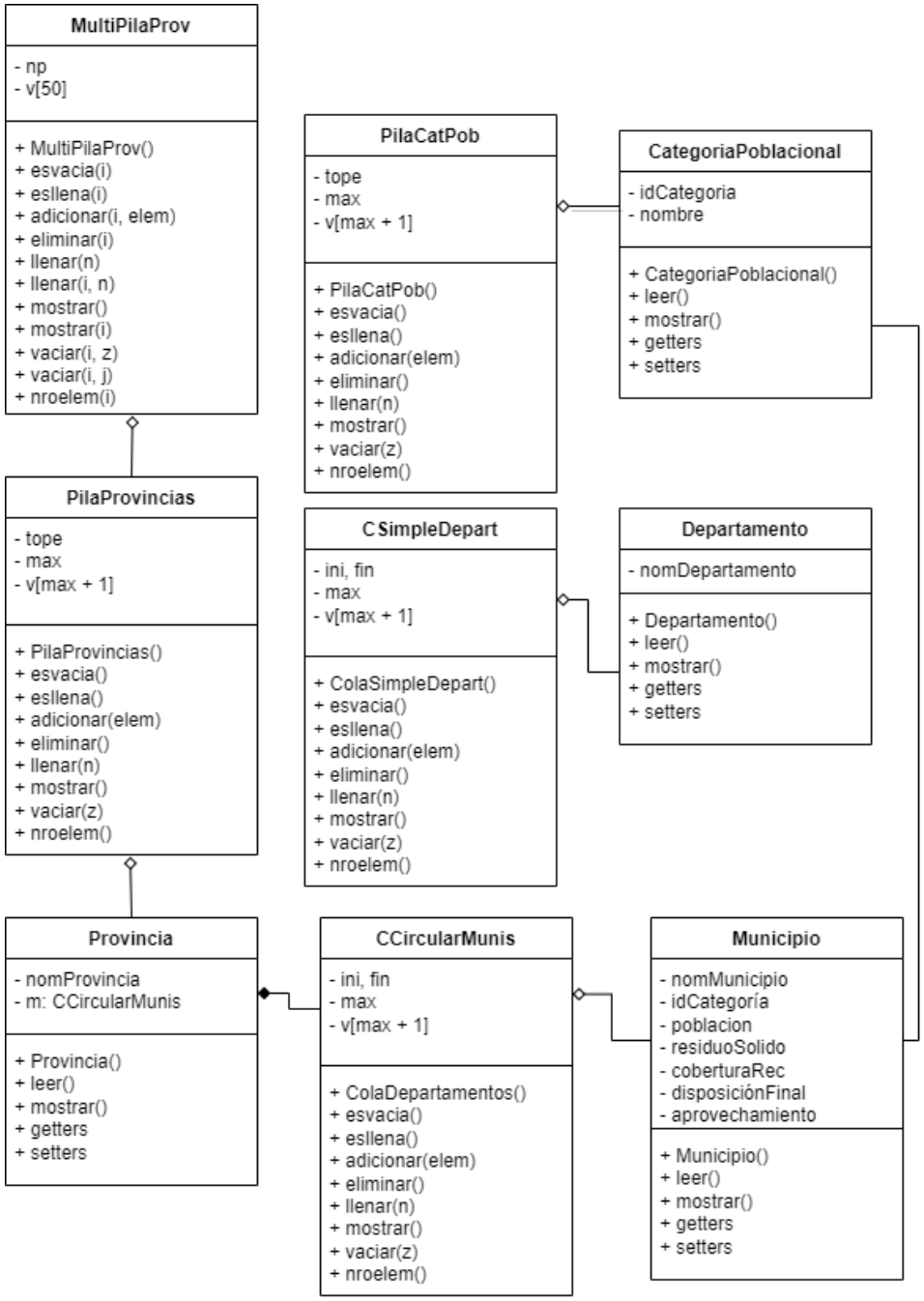
1. **(5 pts)** Elaborar el diagrama de clases con sus atributos

Asumiendo que las estructuras almacenan información, se pide:

1. **(10 pts)** Calcular el promedio de residuos sólidos generados de cada departamento
2. **(10 pts)** Mostrar cada Departamento y sus municipios con menor población en la categoría con nombre x.

NOTA: Se disponen de los métodos básicos de pilas, colas y Múltiple Cola y de objetos se tiene getters, setters, leer y mostrar

Inciso 0


MultiPilaProv
- np
- v[50]
+ MultiPilaProv()
+ esvacia(i)
+ esllena(i)
+ adicionar(i, elem)
+ eliminar(i)
+ llenar(n)
+ llenar(i, n)
+ mostrar()
+ mostrar(i)
+ vaciar(i, z)
+ vaciar(i, j)
+ nroelem(i)
PilaCatPob
- tope
- max
- v[max + 1]
+ PilaCatPob()
+ esvacia()
+ esllena()
+ adicionar(elem)
+ eliminar()
+ llenar(n)
+ mostrar()
+ vaciar(z)
+ nroelem()
CategoriaPoblacional
- idCategoria
- nombre
+ CategoriaPoblacional()
+ leer()
+ mostrar()
+ getters
+ setters
PilaProvincias
- tope
- max
- v[max + 1]
+ PilaProvincias()
+ esvacia()
+ esllena()
+ adicionar(elem)
+ eliminar()
+ llenar(n)
+ mostrar()
+ vaciar(z)
+ nroelem()
CSimpleDepart
- ini, fin
- max
- v[max + 1]
+ ColaSimpleDepart()
+ esvacia()
+ esllena()
+ adicionar(elem)
+ eliminar()
+ llenar(n)
+ mostrar()
+ vaciar(z)
+ nroelem()
Departamento
- nomDepartamento
+ Departamento()
+ leer()
+ mostrar()
+ getters
+ setters
Provincia
- nomProvincia
- m: CCircularMunis
+ Provincia()
+ leer()
+ mostrar()
+ getters
+ setters
CCircularMunis
- ini, fin
- max
- v[max + 1]
+ ColaDepartamentos()
+ esvacia()
+ esllena()
+ adicionar(elem)
+ eliminar()
+ llenar(n)
+ mostrar()
+ vaciar(z)
+ nroelem()
Municipio
- nomMunicipio
- idCategoría
- poblacion
- residuoSolido
- coberturaRec
- disposiciónFinal
- aprovechamiento
+ Municipio()
+ leer()
+ mostrar()
+ getters
+ setters

Inciso 1

```
numero1(MultiPilaProv a, CSimpleDepart b){
  aux = new CSimpleDepart()
  for i = 1 to a.getNp(){
    suma = 0.0
    w = b.eliminar() // departamento
    aux2 = new PilaProvincias()
    while not a.esvacia(i){
      f = a.eliminar(i) // provincia
      aux2.adicionar(f)
      m = f.getM() // CCircularMunis
      suma = suma + sumaTotalResiduosProvincia(m)
    }
    a.vaciar(i, aux2)
    print(w.getNomDepartamento(), ":", suma / a.nroelem(i))
  }
  b.vaciar(aux)
}

sumaTotalResiduosProvincia(CCircularMunis a){
  suma = 0.0
  aux = new CCircularMunis()
  while not a.esvacia(){
    w = a.eliminar() // municipio
    suma = suma + (w.getResiduoSolido() * w.getPoblacion())
    aux.adicionar(w)
  }
  a.vaciar(aux)
  return suma
}
```

Inciso 2

```
numero2(MultiPilaProv a, CSimpleDepart b, PilaCatPob c, String x){
  aux = new CSimpleDepart()
  for i = 1 to a.getNp(){
    w = b.eliminar() // departamento
    aux.adicionar(w)
    print(w.getNomDepartamento(), ":")
    aux2 = new PilaProvincias()
    while not a.esvacia(i){
      y = a.eliminar(i) // provincia
      aux2.adicionar(y)
      m = y.getM() // CCircularMunis
      idCateg = obtenerCategoria(b, x)
      menorPoblacionCategoriaX(a, idCateg)
    }
    a.vaciar(i, aux2)
  }
  b.vaciar(aux)
}


menorPoblacionCategoriaX(CCircularMunis a, int diCateg){
  menorPoblacion = 1000000000
  n = a.nroelem()
  sw = false // no hay municipio con categ x
  for i = 1 to n{
    f = a.eliminar() // municipio
    if f.getPoblacion() < menorPoblacion and f.getIdCategoria() = idCateg{
      menorPoblacion = f.getPoblacion()
      sw = true
    }
    a.adicionar(f)
  }

  for i = 1 to n{
    f = a.eliminar() // municipio
    if f.getPoblacion() = menorPoblacion and f.getIdCategoria() = idCateg{
      f.mostrar()
    }
    a.adicionar(f)
  }

  if sw = false{
    print("no existe ni un municipio en dicha categoria")
  }
}
```

```
obtenerCategoria(PilaCatPob b, String x){
  idCateg = -1
  aux = new PilaCatPob()
  while not b.esvacia(){
    g = b.eliminar() // CategoriaPoblacional
    aux.adicionar(g)
    if g.getNombre() = x{
      idCateg = g.getIdCategoria()
    }
  }

  b.vaciar(aux)
  return idCateg
}
```