

TDA POLINOMIO

Ing. Mario Milton López Winnipeg

Capitulo 5 TDA Polinomio

5 Polinomio

5.1 Descripción del TDA Polinomio.

5.2 Especificación del TDA Polinomio.

5.3 Ejemplos de uso.

5.4 Implementaciones del TDA Polinomio.

5.4.1 Implementación con vectores.

5.4.2 Implementación con apuntadores

5.4.3 Implementación basada en el TDA Lista.

5.1 Descripción del TDA Polinomio

- Una polinomio es un caso particular de lista en el cual los terminos se ponen cumpliendo las reglas definidas por la matematica.
- El polinomio a reprecentar tiene como coeficiente y exponente de cada termino valores enteros

$$7x^4 + 4x^1 + 5$$

5.2 Especificación del TDA Cola

Especificación Informal

- Polinomio = TDA con operaciones crea, escero grado, poner_termino, coeficiente, sumar, multiplicar, restar, igual, numero_terminos, Exponente
- DESCRIPCIÓN:
 - Los valores del TDA Polinomio son coeficientes y grados del tipo Entero. El polinomio es mutable cuando se pone terminos o sumar, restar, multiplicar polinomios.
- OPERACIONES:
 - **crea()** devuelve (P:Polinomio)
 - efecto: Devuelve la polinomio cero P.

5.2 Especificación del TDA Polinomio

- **EsCero**(P: Polinomio) devuelve (booleano)
 - efecto: Devuelve cierto si P es la polinomio sin terminos
- **Grado**(P:Polinomio) devuelve (Grado del Polinomio)
 - requerimientos: El polinomio es no cero.
 - efecto: Devuelve valor que indica grado polinomio
- **coeficiente**(P:Polinomio,Exp : Entero)
 - requerimiento: Polinomio no Cero
 - efecto: Devuelve el Coeficiente que corresponde al termino con exponente Exp.

5.2 Especificación del TDA polinomio

- **sumar(P1,P2:Polinomio; ES P3: Polinomio)**
 - requerimientos: Dos polinomios P1 y P2
 - efecto: Retorna el P3 como resultado de la suma
- **restar(P1,P2:Polinomio; ES P3: Polinomio)**
 - requerimientos: Dos polinomios P1 y P2
 - efecto: Retorna el P3 como resultado de la resta
- **multiplicar(P1,P2:Polinomio; P3: Polinomio)**
 - requerimientos: Dos polinomios P1 y P2
 - efecto: Retorna el P3 como resultado de la multiplicacion

5.2 Especificación del TDA polinomio

- **poner_termino**(P:Polinomio; coef, exp : entero)
 - requerimientos: Coeficiente y grado de tipo entero
 - efecto: Modifica polinomio P asignando el termino con coeficiente y exp.
- **numero_terminos**(P:Polinomio)
 - requerimientos: Polinomio
 - efecto: retorna el numero de terminos del polinomio
- **exponente**(P:Polinomio; nroter: entero)
 - requerimientos: NroTermino
 - efecto: retorna el exponenete que corresponde al nroTer (Numero de termino).

5.2 Especificación del TDA Polinomio

Especificación Formal

Tipo: Polinomio (Termino)

□ *Sintaxis:*

- `crea` → Polinomio
- `escero(polynomio)` → booleano
- `Poner_termino(polynomio,coef,exp)` → Polinomio
- `Grado(Polinomio)` → Entero
- `Coeficiente(Polinomio,Exp)` → Entero
- `Suma(polynomio,polinomio)` → polinomio
- `resta(polynomio,polinomio)` → polinomio
-

5.2 Especificación del TDA Polinomio

La interface del TDA Polinomio de acuerdo a esta especificación puede definirse de la siguiente forma:

```
publico interface Polinomio
{

completar la presente lamina

} // fin interface polinomio
```

5.3 Ejemplo de uso

```
publico Derivada( Polinomio P)
inicio
  para cada i = 1 hasta p.numero_terminos()
    inicio
      ex = p.exponente(i)
      co = p.coeficiente(ex)
      p.poner_termino(co*-1,exp)
      p.poner_termino(co*ex,ex-1)
    fin
  fin
fin
```

5.4 Implementaciones del TDA Polinomio

- En esta sección mostraremos tres implementaciones para el TDA Polinomio:
 - Implementación con vectores
 - Implementación con apuntadores
 - Implementación basada en el TDA Lista

5.4.1 Implementacion basada en vectores

Definición básica de la clase polinomio cuya implementacion es usando vectores

Clase Polinomio

Atributos

VC, // Coeficientes
VE : Arreglo (MAX) // Exponentes
nt : Entero

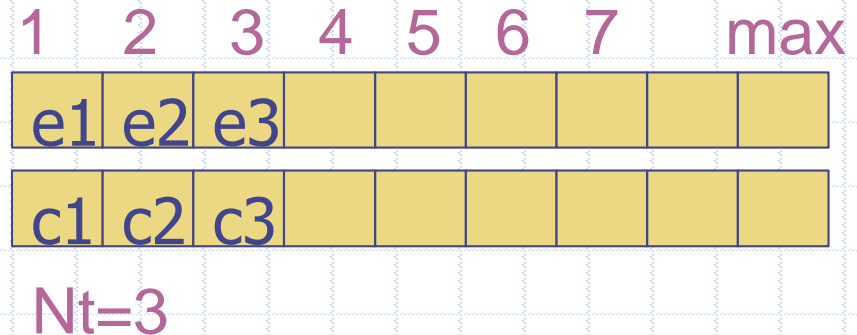
Metodos

Crear()
escero ()
Poner_termino (coe,exp Entero)
Coeficiente(exp:Entero)
Exponente (Nro_ter:Entero)
Grado()
suma (p:polinomio)
resta (p:plinomio())
numero_terminos()

Fin

Constructor Polinomio.Crear

inicio
nt=0
fin



5.4.1 Implementacion basada en vectores

Definición básica de la clase Polinomio cuya implementacion es usando vectores

```
Publico booleano polinomio.escero  ()
Inicio
    retornar (nt=0)
Fin

Publico entero polinomio.grado()
Inicio
    si nt>0 entoces
        max=ve[1]
        para cada i = 1 hasta nt
            si ve[i]>max entoces max=ve[i]
        retornar max
    caso contraio
        // error no existe terminos
fin

Publico entero polinomio.coeficiente(exp : entero)
Inicio
    si exp>=0 y  exp <= grado() entoces
        para cada i = 1 hasta nt
            inicio
                si ve[i] = exp entoces retornar vc[i]
            fin
        // error no existe termino con ese exponente
Fin
```

5.4.1 Implementacion basada en vectores

Definición básica de la clase Polinomio cuya implementación es usando vectores

```
Publico booleano polinomio.poner_termino (coef,exp : entero)
```

```
Inicio
```

```
    lug = // Existe exponente (exp) en la estructura revisando vector ve  
    si lug<>-1 entoces
```

```
        vc[lug]=vc[lug]+coef
```

```
        si vc[lug]= 0 entoces
```

```
            // desplazar 1 elemento hacia la posicion lug
```

```
            nt = nt -1
```

```
        caso contrario
```

```
            nt = nt +1
```

```
            vc[nt]=coef
```

```
            ve[nt]=exp
```

```
Fin
```

```
Publico polinomio.suma(p1, p2 : polinomio)
```

```
Inicio
```

```
    para cada i = 1 hasta p1.numero_terminos
```

```
        inicio
```

```
            ex= p1.exponente(i)
```

```
            co=p1.coeficiente(ex)
```

```
            poner_termino(co,ex)
```

```
        fin
```

```
    para cada i = 1 hasta p2.numero_terminos
```

```
        inicio
```

```
            poner_termino(p1.coeficiente(p2.exponente(i)), p2.exponente(i))
```

```
        fin
```

```
Fin
```

5.4.2 Implementación con apunadores

Definición básica de la clase Polinomio implementada usando punteros:

Tipo de dato

Nodo

```
Coef      Entero,  
Exp       Entero,  
Sig       Puntero a Nodo  
// fin definicion
```

Direccion Puntero a espacio de memoria de tipo Nodo

Clase polinomio

Atributos

Nt Entero // Numero de Polinomio

Terminos

PtrPoli Direccion

Metodos

..... • •

5.4.2 Implementación con apuntadores

```
publico Polinomio.Crear()
    inicio
        ptr_poli=nulo
        nt = 0
    fin

publico Polinomio.poner_termino( coef, exp : Entero)
    inicio
        existe = // Buscar direccion de nodo con grado exp
        si existe = nulo
            entonces
                aux = M.New_Espacio('coef,exp,sig')
                si aux <> nulo entonces
                    m.ponerdato(aux,'->coef',coef)
                    m.ponerdato(aux,'->exp',exp)
                    m.ponerdato(aux,'->sig',prt_poli)
                    Ptr_Poli = Aux
                    nt =nt +1
                    caso contrario//error /espacio memoria
                        caso contrario
                            m.ponerdato(existe,'->coef',m.obtenerdato(existe,'->coef')+ coef)
                            // analizar si coef se pone en 0 ???
            fin
        fin
```


5.4.3 Implementación con lista

Definición básica de la clase polinomio cuya implementacion es usando listas

Clase Polinomio

Atributos

Pol: Lista

// Contiene Coeficiente,grado,coeficiente,grado.....

Metodos

Crear()

escero ()

Poner_termino(coe,exp Entero)

Coeficiente(exp:Entero)

Exponente(Nro_ter:Entero)

Grado()

suma(p:polinomio)

resta(p:plinomio())

numero_terminos()

Fin

Constructor Polinomio.Crear

inicio

// pol.crear

fin

COMPLETAR
IMPLEMENTACION

Practico1

Implementar los metodos pertenecientes a la clase Polinomio

Mostrar: Muestra el polinomio en el siguiente formato

Coef X^{Grado} Ej: $3X^2$

Evaluar(x): Función que retorna el resultado de reemplazar X en el polinomio.

PROYECTO 1 (SMemoria)

- 1) Crear memoria
- 2) Pedir espacio
- 3) Liberar espacio
- 4) Crear polinomio
- 5) Poner Terminio
- 6) Mostrar Polinomio
- 7) Derivar
- 8) Salir

PROYECTO 2 (Vectores)

- 1) Crear polinomio
- 2) Poner Terminio
- 3) Mostrar Polinomio
- 4) Derivar
- 5) Salir

PROYECTO 3 (Punteros)

- 1) Crear polinomio
- 2) Poner Terminio
- 3) Mostrar Polinomio
- 4) Derivar
- 5) Salir

Datos Entrada
Coef, Grado Datos Enteros

PROYECTO 4 (Lista)

- 1) Crear polinomio
- 2) Poner Terminio
- 3) Mostrar Polinomio
- 4) Derivar
- 5) Salir

PROYECTO 4 (Lista con SMemoria)

- 1) Crear polinomio
- 2) Poner Terminio
- 3) Mostrar Polinomio
- 4) Derivar
- 5) Salir