

Ejercicio

- Clasificación -

► Evolución Histórica

- Lenguajes de 2da generación
- Lenguajes de 5ª generación

► Manera de ejecutar

- Lenguajes interpretados
- Lenguajes compilados

► Forma de abordar la

Tarea

- Lenguajes Imperativos
- Lenguajes Declarativos

► Paradigma de programación

- Lenguajes Orientado a Objetos
- Lenguaje Funcionales

9 de febrero del
2018

- Marieta Villegas
Alfonso

- Grupo 1

(7:00 - 9:00)

Ejercicio 1

10

Ejercicios

- 14 de febrero de 2018

- Marrieta Villegas

Al fondo

- Grupo 1 (7:00-9:00)

- Exercício 2

a) 1/1000 101 1/0011/000000000000/0000/000

b) $35.\overline{375} \rightarrow 35.375$

c) -145.44

a) $1 \mid 10001011 \mid 0011 \dots$

negative \times

$$\begin{array}{r} 139 \\ - 127 = E \\ \hline 12 \end{array}$$

$$\left. \begin{array}{l} 2^{-3} = \frac{1}{8} \\ 2^{-4} = \frac{1}{16} \end{array} \right\} \cdot 1875 + 1$$

$$\therefore 2^{12} = 4096$$

$$\therefore (-1)(4096)(1.1875) = -4864 //$$

b) 35.375

3	2	1	6	8	4	2	1
1	0	0	0	1	1		

100011
altos = 5

Número Final

0 | 10000100 | 0001011
↑ ↑
Positive Exp

$$E + S; 127 + 5 = 132$$

128	64	32	16	8	4	2	1
1	0	0	0	0	1	0	0

8 bits exponential

$$\begin{array}{r} .375 \times 2 = .75 \\ .75 \times 2 = 1.5 \\ .5 \times 2 = 1.0 \end{array}$$

Número
final

0 | 10000100 | 000110110000000000000000

↑ ↑
Positivo Exp

c) -145.44

Es negativo = 1

$$145 = \begin{array}{c|c|c|c|c|c|c|c} 128 & 64 & 32 & 16 & 8 & 4 & 2 & 1 \\ \hline 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{array}$$

1.0010001
7 saltos

E + 7 = 134

$$134 = \begin{array}{c|c|c|c|c|c|c|c} 128 & 64 & 32 & 16 & 8 & 4 & 2 & 1 \\ \hline 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{array} \text{Exp} \quad 110$$

.44 x 2 = .88

.88 x 2 = 1.76

.76 x 2 = 1.52

.52 x 2 = 1.04

.04 x 2 = .08

.08 x 2 = .16

... Lo mundo ☹ perdon...

Segundo envío

Nombre Murrieta VillegasGrupo: 1

1.- En las líneas de código que se indica escribe "C" si la afirmación es correcta o "I" si es incorrecta o generaría un error de compilación

```
int arreglo[8];
int var1 = 0;
float b = 5.0;
int a = 2;
arreglo[3] = 1;
arreglo[var1] = 4;
arreglo[a++] = 2;
arreglo[b] = 3;
arreglo[a*4] = 6;
arreglo[a*2] = 7;
```

C
C
C
I
C
C

← $[a*4] \sim [8]$ ← se pasa

9.2

2.- Realiza la representación gráfica del arreglo resultante después de ejecutar las instrucciones siguientes:

```
int arreglo[4][3] = {2, 4, 6, 3, 6, 9, 8, 10, 12, 12, 15, 18};
arreglo[1][2] = 110;
arreglo[0][1] = 30;
arreglo[3][3] = 45;
arreglo[2][1] = 199;
arreglo[0][2] = 300
```

0	2	30	6
1	3	6	110
2	8	199	12
3	12	15	300

2	4	6
3	6	9
8	10	12
12	15	18

Tabla final

3.- Indica la salida del siguiente código

```
void main() {
    int a[3][4][2] = {4, 8, 12, 16, 20, 24, ..., 76, 80, 84, 88, 92, 96};
    printf("\n Primer valor: %d \n", a[1][1][1]);
    printf("\n Segundo valor: %d \n", a[0][3][1]);
    printf("\n Tercer valor: %d \n", a[2][2][1]);
    printf("\n cuarto valor: %d \n", a[2][0][0]);
    system("pause");
}
```

Salida

Primer Valor: 48
Segundo valor: 32
Tercer valor: 88
Cuarto valor: 68

Nota: No di el salto de línea debido a \n

0	4	8
1	12	16
2	20	24
3	28	32

0	36	40
1	44	48
2	52	56
3	60	64

0	68	72
1	76	80
2	84	88
3	92	96

- Ejercicio - 4
 - 16 de febrero 2018
 - Marieta Villegas Alfonso
 - Grupo 1 (7:00 - 9:00)

a) procedure mystery

a: integer;

b: integer;

procedure enigma(x,y)

begin

y = y + b;

x = b + x;

b = x + b;

a = y;

end enigma

begin

a = 2 ; b = 7;

enigma (a,b);

write (a); write(b);

end mystery;

SALIDAS

► Paso por Valor

a = 2

b = 7 //

► Paso por Referencia

x = a

y = b

y = y + b ; y = 7 + 7 ; y = 14

x = 7 + a ; x = 7 + 2 ; x = 9

b = 9 + 7 ; b = 16

y = a = 14

∴ a = 14

b = 16 //

b) Indicar los valores que se piden

3: int a=10, b=20, c=30, d=100, r1, r2, r3;

4: int *p1, *p2, *p3, *p4;

p1 = &a; p2 = &b; p3 = &c; p4 = p1;

r1 = *p1 + *p2; r1 = 10 + 20; r1 = 30

r2 = *p3 * *p4; r2 = 30 * 10; r2 = 300

r3 = *p2 * *p4; r3 = 20 * 10; r3 = 200

*p1 = 44;

*p2 = 23;

*p3 = d;

d? a = 44 /

d? b = 23 /

c = 100 /

r1 = 30 /

r2 = 300 /

r3 = 200 /

- Ejercicio 5 -

- 28 de febrero de 2018

- Murrieta Villegas Alfonso

- Grupo 1 (7:00 - 9:00):

95

```
#include <stdio.h>
```

```
struct direccion {
```

```
    char calle[40];
```

```
    int numero;
```

```
    char colonia[40];
```

```
    int cp;
```

```
} typedef struct direccion Direccion;
```

```
struct alumno {
```

```
    int numCuenta;
```

```
    char nombre[20];
```

```
    char apellido[20];
```

```
    float promedio;
```

```
    struct Direccion;
```

```
} typedef struct alumno Alumno;
```

```
main() {
```

```
    Alumno alumno1 *ap; ap = &alumno1;
```

```
    alumno1.numCuenta = 31504;
```

```
    alumno1.nombre = "Alfonso";
```

```
    alumno1.apellido = "Murrieta";
```

```
    alumno1.direccion.char = "Calle...";
```

```
    alumno1.direccion.numero = "65717781";
```

```
    alumno1.direccion.colonia = "colonia";
```

```
    alumno1.direccion.cp = "11031";
```

```
    ap->direccion.calle = "Calle 2";
```

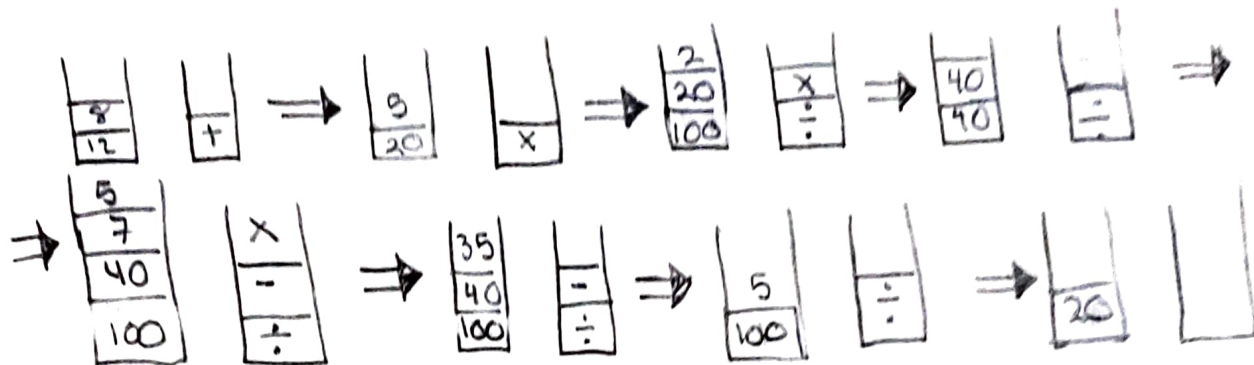
```
}
```

identificador

- Ejercicio 6
 - 21 de Marzo de 2018
 - Horreba Villegas Alfonso
 - Grupo 1 (7:00-9:00)

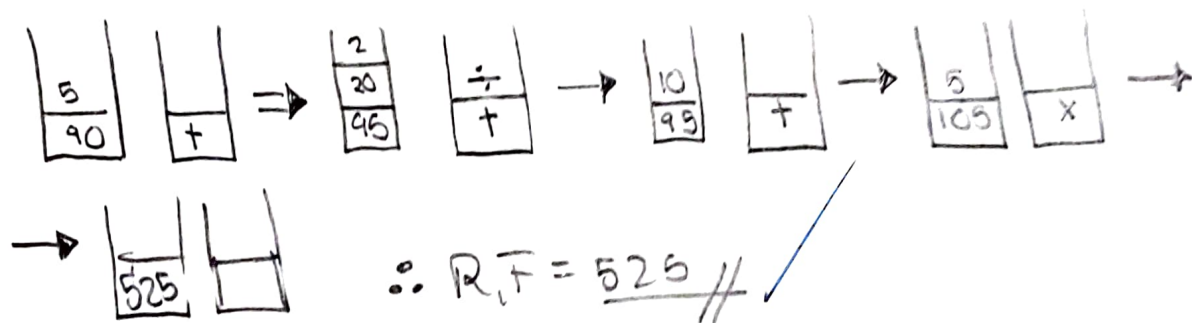
10

$$I] \left((12+8) \times 5 \right) \div \left((20 \times 2) - (7 \times 5) \right)$$



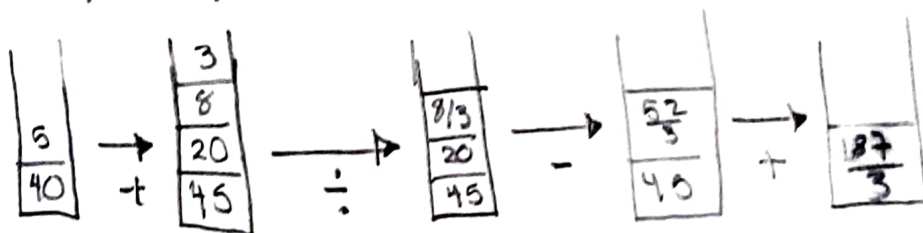
$$\therefore R.F. = \underline{20} //$$

$$II] \left((90+5) + (20 \div 2) \right) \times 5$$



$$\therefore R.F. = \underline{525} //$$

$$III] 40, 5, +, 20, 8, 3, \div, -, +$$



$$\frac{187}{3} = 62.33$$

$$\therefore R.F. = \underline{\frac{187}{3}} \approx 62.333 //$$

7.5

Ejercicio 7
- 4 de Abril de 2018
- Morrieta Villegas Alfonso
- Grupo 1

- a) Escribir el código de búsqueda en una lista ligada simple
- b) Escribir el código para insertar un nodo en una lista ligada simple en la posición 3

```
a) Nodo tmp;;  
tmp = head;  
while (tmp.next != 10) {  
    tmp = tmp.next;  
}
```

Handwritten notes: "nodo" points to tmp, "int" points to 10.

//Falta XD

```
b) Nodo nodo3;  
nodo3.inf = 10;  
Nodo tmp;  
tmp.inf = 6; tmp.next = head;  
head.next = head; ?  
head.next.next = nodo3;
```

sirve para
desplazar
todos una
posición y
en el tercero
que es
head.next.next
le insertamos
directamente el
nodo 3 (creado).

7

► Murrieta Villegas Alfonso
► EDA 1

► Ejercicio 8

a) ► Código para invertir los elementos de una lista simple. (caso de 3 elementos)

// se crea lista

```
nodo tail;  
tail.next = head.next;  
head.next = head;  
head.next = null
```

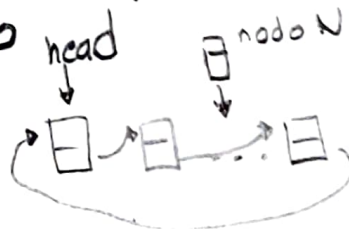
// Están invertidos
falta generalizar



Para generalizar
Meter en ciclo

► Insertar un elemento en la posición n de una lista circular

```
int size; // definir tamaño  
nodo head;  
nodo nodoN;  
nodoN.info = 7;  
posición = n; // aquí el n es  
cualquier número
```



```
if (n != size)
```

```
nodoN.next = head.next.next
```

```
head.next.next = nodoN
```

else

```
print (Ingrese una posición dentro de la  
cantidad elementos)
```

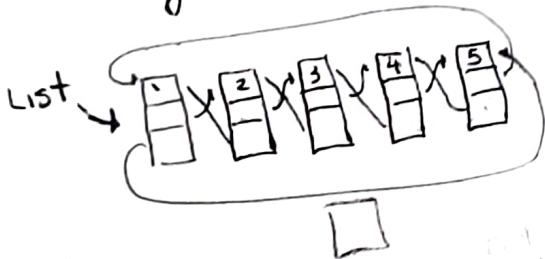
Meter en
ciclo

// Otra alternativa para comparar es que dentro de los
ciclos se considere el size como límite de la lista

- Murrieta Villegas Alfonso
- EDA 1
- Edgar Tista
- Ejercicio 9

7

- ✓ Escribir el pseudocódigo para
- eliminar el nodo 3
 - Agregar un nuevo nodo entre 4 y 5



* num es la posición a borrar 4/

```

Node tmp = head;
tmp = list.next.next;
list.next = list.prev.prev;
tmp.prev = list.next.next;
borrar i esimo (Nodo 3)
list.next = tmp.prev;
list.prev.prev = tmp.next;

```



Nodo Nuevo = head;

! Pendiente ?

6 (En caso correcto)

Problema de la mochila

Ejercicio 10

Murieta Villegas Alfonso

Grupo 1

EDA-I

9.5

DATOS

Peso	10	20	30	45	50	60	70
Costo	30	35	55	45	65	80	60
Proporción de costo / peso	3	1.75	1.83	1	1.3	1.333	0.857142

Capacidad = 190 [kg]

1] Mochila Normal

Primer Caso (Más ligero)

Peso $10 + 20 + 30 + 45 + 80 = 185$ [kg] ✓
 Valor $30 + 35 + 55 + 45 = \$165$

Segundo caso (El más caro)

Peso $= 60 + 50 + 30 = 140$ [kg] *aun cabe el de 10kg*
 Valor $= 80 + 65 + 55 = \$200$

Tercer caso

Menor Valor = 3, 1.83, 1.75, 1.333 → Ya no caben
 Peso $= 10 + 30 + 20 + 60 = 120$ [kg]
 Valor $= 30 + 55 + 35 + 80 = \$200$ ✓

2] Mochila Fraccional

Primer caso (Más ligero)

Peso $= 10 + 20 + 30 + 45 + 45$ (de 50) $= 150$ [kg] ✓
 Valor $= 30 + 35 + 55 + 45 + 4 \cdot 58.5 = 223.5$ [kg] ✓

Segundo caso (Más caro)

Peso $= 60 + 50 + 40$ (de 70) $= 150$ [kg]
 Valor $= 80 + 65 + 39.28 = 179.28$ \$ ✓

Tercer caso

Peso $= 10 + 30 + 20 + 60 + 30$ (de 50) $= 150$ [kg]
 Valor $= 30 + 55 + 35 + 80 + 39 = 239$ \$ ✓

50 = 65
30 = 39

• Morrieta Villegas Alfonso

• Grupo 1

• EDA-I

Actividad 11
(Ejercicio)

9

Funciones recursivas

- ① Imprimir números impares entre 0 y n de manera descendente
- ② Que determine el tamaño de una lista ligada simple

```
① int sumaImpares (int n) {  
    if (n == 1)  
        return 1;  
    else
```

↳ caso base n es igual
1/0/1
1/1 es el caso base de
los impares

```
    return sumaImpares (n-1) + n; ← Parte  
    }                               recursiva
```

```
② int TamañoLista (Lista lista1) {  
    if (tmp.next == null) {  
        return tamaño;  
    } else
```

↳ El caso base es el
último elemento?

```
    tamaño = (tamaño + 1);
```

```
    tmp = tmp.next;
```

```
    return primer (Lista lista1)
```

```
    return (Lista lista1)
```

```
}
```

• Murrieta Villegas Alfonso
 • (Actividad) 12
 Ejercicio

• EDA 1

• Grupo 7:00-9:00 a.m. 1

9

1) $7 \rightarrow O(1)$ Constante

2) $\log(\log(n)) \rightarrow$ Logaritmico

3) $\log(n) \rightarrow$ Logaritmico

4) $\sqrt[n]{n} = n^{1/n} \rightarrow$ Polinomial X

5) $\sqrt{n} = n^{1/2} \rightarrow$ Polinomial X

6) $(\log(n))^4 = \text{contar } \log(n) \rightarrow$ Linear-logaritmico logaritmico ✓

7) $1714n \rightarrow$ Linear

8) $n^2 \rightarrow n^c$ Polinomial cuadrática

9) $100n^3 \rightarrow n^c$ Polinomial cúbica

10) $10n^4 \rightarrow n^c$ Polinomial 4º grado

11) $(\frac{3}{2})^n \rightarrow c^n$ Exponencial

12) $2^n \rightarrow c^n$ Exponencial

13) $8n! \rightarrow n!$ Factorial

Ordenada

- ① 1) 7
- ② 2) $\log(\log n)$
- ③ 3) $\log(n)$
- ④ 4) $\log(n)^4$
- ⑤ 5) $1714(n)$
- ⑥ 6) \sqrt{n}
- ⑦ 7) n^2
- ⑧ 8) $100n^3$
- ⑨ 9) $10n^4$
- ⑩ 10) $10n^4$

- ⑪ 11) $(\frac{3}{2})^n$
- ⑫ 12) 2^n
- ⑬ 13) $8n!$

Grupo: 1

EDA: 1

Murieta Villegas Alfonso

Edgar Tista

Ejercicio 13

9

① ~~funcion(1)~~
~~for (cnt=1; ~~i=1~~; cnt++)~~
~~for (~~

funcion 1() {
for (cnt=0; i=1; i<=n; i++)
for (j=1; j<=n; j++)
cnt++;
}
 $\therefore O(n)^2$
Polinomio Cuadrático

② funcion2(1)
for (cnt3=0, i=1; i<=n; ~~i*=2~~)
for (j=1; j<=n; j++)
cnt3++; int a=0; int b=10;

Complejidad

$\therefore 3 \cancel{O(n)^2}$ Polinomio Cuadrático

③ funcion3(int n) {
if (n <= 0)
return 1;
else
return 1 + funcion3(n-8);
}

Complejidad

$\therefore O(n)$ Lineal

$\therefore \Omega \rightarrow O(n)$ Lineal
 $\therefore \Theta = O(n)$

① Argumento es n^2 debido a que tiene dos ciclos for anidados de 1 a n o sea un n para cada n

② Pese esta el " $i*=2$ " de igual forma seguiría siendo

③ solo se llama una vez la función recursiva.