

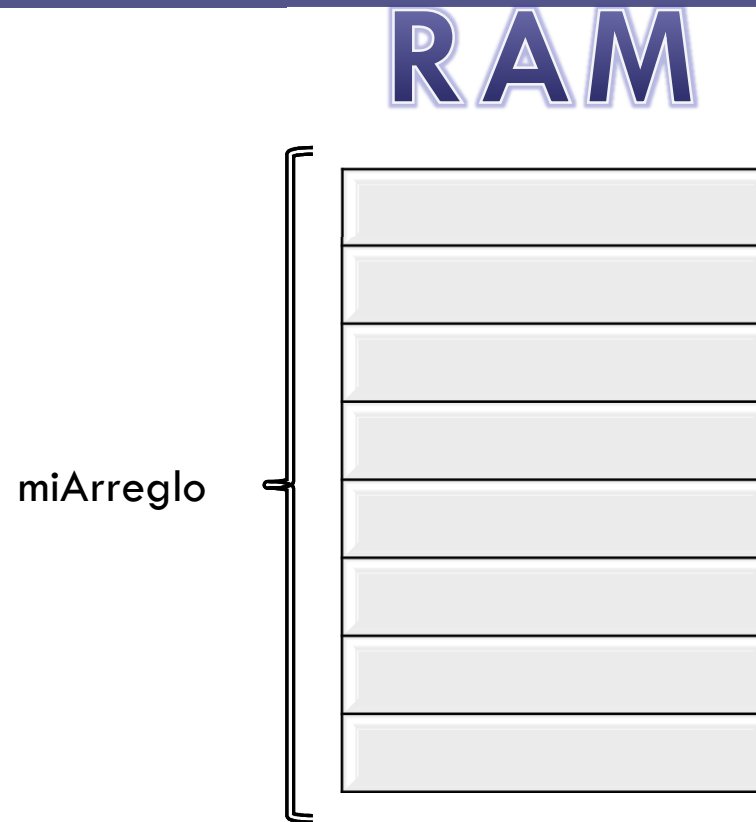
ARREGLOS

25/10/2013

¿Qué es un arreglo?

2

- Variable que hace referencia a varias posiciones de memoria.
- Cada posición se identifica con un índice.
- El índice comienza en 0.



¿Qué es un arreglo?

3

- Variable que hace referencia a varias posiciones de memoria.
- Cada posición se identifica con un índice.
- El índice comienza en 0.

miArreglo

RAM

miArreglo[0]	
miArreglo[1]	
miArreglo[2]	
miArreglo[3]	
miArreglo[4]	
miArreglo[5]	
miArreglo[6]	
miArreglo[7]	

¿Qué es un arreglo?

4

□ Declaración:

```
int main (){  
    Tipo de dato { int miArreglo[8];  
    system("Pause");  
    return 0;  
} // Fin main
```

miArreglo

RAM

miArreglo[0]	
miArreglo[1]	
miArreglo[2]	
miArreglo[3]	
miArreglo[4]	
miArreglo[5]	
miArreglo[6]	
miArreglo[7]	

¿Qué es un arreglo?

5

□ Declaración:

```
int main (){  
    int miArreglo[8];  
    system("Pause");  
    return 0;  
} // Fin main
```

Corchetes para
indicar que es
arreglo.

miArreglo

RAM

miArreglo[0]	
miArreglo[1]	
miArreglo[2]	
miArreglo[3]	
miArreglo[4]	
miArreglo[5]	
miArreglo[6]	
miArreglo[7]	

¿Qué es un arreglo?

6

□ Declaración:

```
int main (){  
    Nombre del arreglo  
    int miArreglo[8];  
  
    system("Pause");  
    return 0;  
} // Fin main
```

RAM

miArreglo

miArreglo[0]	
miArreglo[1]	
miArreglo[2]	
miArreglo[3]	
miArreglo[4]	
miArreglo[5]	
miArreglo[6]	
miArreglo[7]	

¿Qué es un arreglo?

7

□ Asignación de memoria:

```
int main (){  
    int miArreglo[8];  
    system("Pause");  
    return 0;  
} // Fin main
```

Asignación de 8 posiciones
de memoria para valores de
tipo entero.

miArreglo

RAM

miArreglo[0]	
miArreglo[1]	
miArreglo[2]	
miArreglo[3]	
miArreglo[4]	
miArreglo[5]	
miArreglo[6]	
miArreglo[7]	

¿Qué es un arreglo?

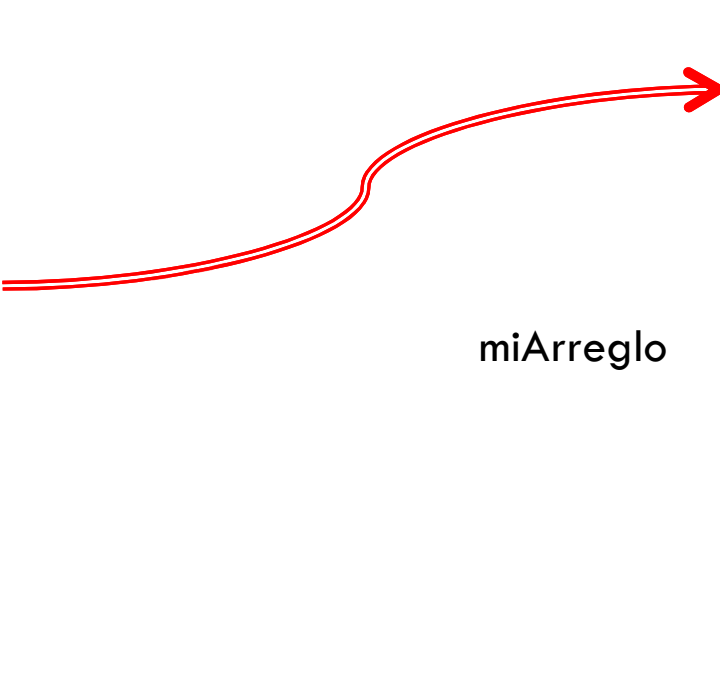
8

□ Asignación de memoria:

```
int main (){  
    int miArreglo[8];  
    miArreglo[0] = 5  
    system("Pause");  
    return 0;  
} // Fin main
```

miArreglo

RAM



miArreglo[0]	5
miArreglo[1]	
miArreglo[2]	
miArreglo[3]	
miArreglo[4]	
miArreglo[5]	
miArreglo[6]	
miArreglo[7]	

¿Qué es un arreglo?


9

□ Asignación de memoria:

```
int main (){  
    int miArreglo[8];  
    miArreglo[0] = 5;  
    miArreglo[1] = 10;  
    system("Pause");  
    return 0;  
} // Fin main
```

miArreglo

RAM



miArreglo[0]	5
miArreglo[1]	10
miArreglo[2]	
miArreglo[3]	
miArreglo[4]	
miArreglo[5]	
miArreglo[6]	
miArreglo[7]	

¿Qué es un arreglo?

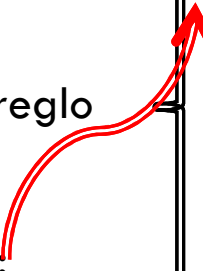
10

□ Asignación de memoria:

```
int main (){  
    int miArreglo[8];  
  
    miArreglo[0] = 5;  
  
    miArreglo[1] = 10;  
  
    miArreglo[2] = miArreglo[0]+ miArreglo[1];  
  
    system("Pause");  
    return 0;  
} // Fin main
```

RAM

miArreglo



miArreglo[0]	5
miArreglo[1]	10
miArreglo[2]	15
miArreglo[3]	
miArreglo[4]	
miArreglo[5]	
miArreglo[6]	
miArreglo[7]	

Ejercicio

11

- ❑ Crear un programa que declare un arreglo llamado "vector" de 10 posiciones.
- ❑ Asignar el valor de 10 a cada posición del arreglo.
- ❑ Mostrar todas las posiciones del arreglo.

vector[0]	10
vector[1]	10
vector[2]	10
vector[3]	10
vector[4]	10
vector[5]	10
vector[6]	10
vector[7]	10
vector[8]	10
vector[9]	10

Código repetitivo.

12

```
int main (){
    int vector [10];

    vector[0] = 10;
    vector[1] = 10;
    vector[2] = 10;
    vector[3] = 10;
    vector[4] = 10;
    vector[5] = 10;
    vector[6] = 10;
    vector[7] = 10;
    vector[8] = 10;
    vector[9] = 10;

    printf ("%d", vector[0]);
    printf ("%d", vector[1]);
    printf ("%d", vector[2]);
    printf ("%d", vector[3]);

    system("Pause");
    return 0;
} // Fin main
```

25/10/2013

Código repetitivo.

13

```
int vector [10];
```

```
vector[0] = 10;  
vector[1] = 10;  
vector[2] = 10;  
vector[3] = 10;  
vector[4] = 10;  
vector[5] = 10;  
vector[6] = 10;  
vector[7] = 10;  
vector[8] = 10;  
vector[9] = 10;
```

```
printf ("%d", vector[0]);  
printf ("%d", vector[1]);  
printf ("%d", vector[2]);  
printf ("%d", vector[3]);  
...
```

```
int vector [10];
```

```
int i = 0;  
while (i<10){  
    vector[i] = 10;  
    i++;  
}
```

```
i = 0;  
while (i<10){  
    printf ("%d", vector[i]);  
    i++;  
}
```

25/10/2013

¿Cómo funciona?

14

```
int main (){  
    int vector [10];  
  
    int i = 0;  
    while (i<10){  
        vector[i] = 10;  
        i++;  
    }  
  
    i = 0;  
    while (i<10){  
        printf ("%d", vector[i]);  
        i++;  
    }  
  
    system("Pause");  
    return 0;  
} // Fin main
```

vector[0]	
vector[1]	
vector[2]	
vector[3]	
vector[4]	
vector[5]	
vector[6]	
vector[7]	
vector[8]	
vector[9]	

¿Cómo funciona?

15

```
int main (){  
  
    int vector [10];  
  
    → int i = 0;  
    while (i<10){  
        vector[i] = 10;  
        i++;  
    }  
  
    i = 0;  
    while (i<10){  
        printf ("%d", vector[i]);  
        i++;  
    }  
  
    system("Pause");  
    return 0;  
} // Fin main
```

vector[0]	
vector[1]	
vector[2]	
vector[3]	
vector[4]	
vector[5]	
vector[6]	
vector[7]	
vector[8]	
vector[9]	
i	0

25/10/2013

¿Cómo funciona?

16

```
int main (){  
    int vector [10];  
  
    int i = 0;  
    while (i<10){  
        vector[i] = 10;  
        i++;  
    }  
  
    i = 0;  
    while (i<10){  
        printf ("%d", vector[i]);  
        i++;  
    }  
  
    system("Pause");  
    return 0;  
} // Fin main
```



true


vector[0]	
vector[1]	
vector[2]	
vector[3]	
vector[4]	
vector[5]	
vector[6]	
vector[7]	
vector[8]	
vector[9]	
i	0

25/10/2013

¿Cómo funciona?

17

```
int main (){  
  
    int vector [10];  
  
    int i = 0;  
    while (i<10){  
        vector[i] = 10;  
        i++;  
    }  
  
    i = 0;  
    while (i<10){  
        printf ("%d", vector[i]);  
        i++;  
    }  
  
    system("Pause");  
    return 0;  
} // Fin main
```



vector[0]	10
vector[1]	
vector[2]	
vector[3]	
vector[4]	
vector[5]	
vector[6]	
vector[7]	
vector[8]	
vector[9]	
i	0

25/10/2013

¿Cómo funciona?

18

```
int main (){  
    int vector [10];  
  
    int i = 0;  
    while (i<10){  
        vector[i] = 10;  
        i++;  
    }  
  
    i = 0;  
    while (i<10){  
        printf ("%d", vector[i]);  
        i++;  
    }  
  
    system("Pause");  
    return 0;  
} // Fin main
```



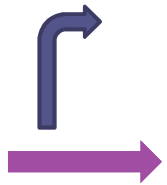
vector[0]	10
vector[1]	
vector[2]	
vector[3]	
vector[4]	
vector[5]	
vector[6]	
vector[7]	
vector[8]	
vector[9]	
i	1

25/10/2013

¿Cómo funciona?

19

```
int main (){  
  
    int vector [10];  
  
    int i = 0;  
    while (i<10){  
        vector[i] = 10;  
        i++;  
    }  
  
    i = 0;  
    while (i<10){  
        printf ("%d", vector[i]);  
        i++;  
    }  
  
    system("Pause");  
    return 0;  
} // Fin main
```



vector[0]	10
vector[1]	
vector[2]	
vector[3]	
vector[4]	
vector[5]	
vector[6]	
vector[7]	
vector[8]	
vector[9]	
i	1

25/10/2013

¿Cómo funciona?

20

```
int main (){  
    int vector [10];  
  
    int i = 0;  
    while (i<10){  
        vector[i] = 10;  
        i++;  
    }  
  
    i = 0;  
    while (i<10){  
        printf ("%d", vector[i]);  
        i++;  
    }  
  
    system("Pause");  
    return 0;  
} // Fin main
```



true


vector[0]	10
vector[1]	
vector[2]	
vector[3]	
vector[4]	
vector[5]	
vector[6]	
vector[7]	
vector[8]	
vector[9]	
i	1

25/10/2013

¿Cómo funciona?

21

```
int main (){  
  
    int vector [10];  
  
    int i = 0;  
    while (i<10){  
        vector[i] = 10;  
        i++;  
    }  
  
    i = 0;  
    while (i<10){  
        printf ("%d", vector[i]);  
        i++;  
    }  
  
    system("Pause");  
    return 0;  
} // Fin main
```



vector[0]	10
vector[1]	10
vector[2]	
vector[3]	
vector[4]	
vector[5]	
vector[6]	
vector[7]	
vector[8]	
vector[9]	
i	1

25/10/2013

¿Cómo funciona?

22

```
int main (){  
    int vector [10];  
  
    int i = 0;  
    while (i<10){  
        vector[i] = 10;  
        i++;  
    }  
  
    i = 0;  
    while (i<10){  
        printf ("%d", vector[i]);  
        i++;  
    }  
  
    system("Pause");  
    return 0;  
} // Fin main
```

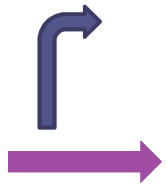


vector[0]	10
vector[1]	10
vector[2]	
vector[3]	
vector[4]	
vector[5]	
vector[6]	
vector[7]	
vector[8]	
vector[9]	
i	2

¿Cómo funciona?

23

```
int main (){  
  
    int vector [10];  
  
    int i = 0;  
    while (i<10){  
        vector[i] = 10;  
        i++;  
    }  
  
    i = 0;  
    while (i<10){  
        printf ("%d", vector[i]);  
        i++;  
    }  
  
    system("Pause");  
    return 0;  
} // Fin main
```



vector[0]	10
vector[1]	10
vector[2]	
vector[3]	
vector[4]	
vector[5]	
vector[6]	
vector[7]	
vector[8]	
vector[9]	
i	2

¿Cómo funciona?

24

```
int main (){  
    int vector [10];  
  
    int i = 0;  
    while (i<10){  
        vector[i] = 10;  
        i++;  
    }  
  
    i = 0;  
    while (i<10){  
        printf ("%d", vector[i]);  
        i++;  
    }  
  
    system("Pause");  
    return 0;  
} // Fin main
```

true


vector[0]	10
vector[1]	10
vector[2]	
vector[3]	
vector[4]	
vector[5]	
vector[6]	
vector[7]	
vector[8]	
vector[9]	
i	2

25/10/2013

¿Cómo funciona?

25

```
int main (){  
  
    int vector [10];  
  
    int i = 0;  
    while (i<10){  
        vector[i] = 10;  
        i++;  
    }  
  
    i = 0;  
    while (i<10){  
        printf ("%d", vector[i]);  
        i++;  
    }  
  
    system("Pause");  
    return 0;  
} // Fin main
```



vector[0]	10
vector[1]	10
vector[2]	10
vector[3]	
vector[4]	
vector[5]	
vector[6]	
vector[7]	
vector[8]	
vector[9]	
i	2

25/10/2013

¿Cómo funciona?

26

```
int main (){  
    int vector [10];  
  
    int i = 0;  
    while (i<10){  
        vector[i] = 10;  
        i++;  
    }  
  
    i = 0;  
    while (i<10){  
        printf ("%d", vector[i]);  
        i++;  
    }  
  
    system("Pause");  
    return 0;  
} // Fin main
```



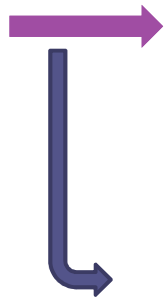
vector[0]	10
vector[1]	10
vector[2]	10
vector[3]	
vector[4]	
vector[5]	
vector[6]	
vector[7]	
vector[8]	
vector[9]	
i	3

25/10/2013

¿Cómo funciona?

27

```
int main (){  
    int vector [10];  
  
    int i = 0;  
    while (i<10){  
        vector[i] = 10;  
        i++;  
    }  
  
    i = 0;  
    while (i<10){  
        printf ("%d", vector[i]);  
        i++;  
    }  
  
    system("Pause");  
    return 0;  
} // Fin main
```



false

vector[0]	10
vector[1]	10
vector[2]	10
vector[3]	10
vector[4]	10
vector[5]	10
vector[6]	10
vector[7]	10
vector[8]	10
vector[9]	10
i	10

Ejercicio

28

- ❑ Crear un arreglo de 100 posiciones.
- ❑ Llenar el arreglo con la tabla del 2.
- ❑ Mostrar el arreglo en pantalla.

vector[0]	0
vector[1]	2
vector[2]	4
vector[3]	6
vector[4]	8
vector[5]	10
vector[6]	12
vector[7]	14
vector[8]	16
vector[9]	18
...	...

Ejercicio

29

- ❑ Crear un arreglo de 100 posiciones.
- ❑ Llenar el orden inverso al índice.
- ❑ Mostrar el arreglo en pantalla.

vector[0]	99
vector[1]	98
vector[2]	97
vector[3]	96
vector[4]	95
vector[5]	94
vector[6]	93
vector[7]	92
vector[8]	91
vector[9]	90
...	...

Ejercicio

30

1

- Datos
 - ▣ $A = [3, 5, 6, 8, 4, 7, 8, 5, 3, 1]$
 - ▣ $B = [3, 4, 6, 8, 9, 1, 2, 3, 0, 9]$

- Realizar las siguientes operaciones
 - ▣ $A[3] \bmod (B[2]/2)$
 - ▣ $B[A[1]] - A[9]$
 - ▣ $A[0] + A[1+2]$
 - ▣ $A[5] + B[5]$
 - ▣ $(A[3]/B[2])/2$

Ejercicio

31

2

- ☐ Crea un arreglo de 20 posiciones.
- ☐ Asígnale a cada elemento un valor.
- ☐ Calcula el promedio de todos los elementos.
- ☐ Calcula la multiplicación de todos los elementos.