Virrey Morcillo
Instituto de Educación Secundaria

UT 1. POO, EDA y API JAVA.

Tipos Compuestos de Datos.

Cadenas.

Arrays Multidimensionales.

Enumeraciones.

Virrey Morcillo
Instituto de Educación Secundaria

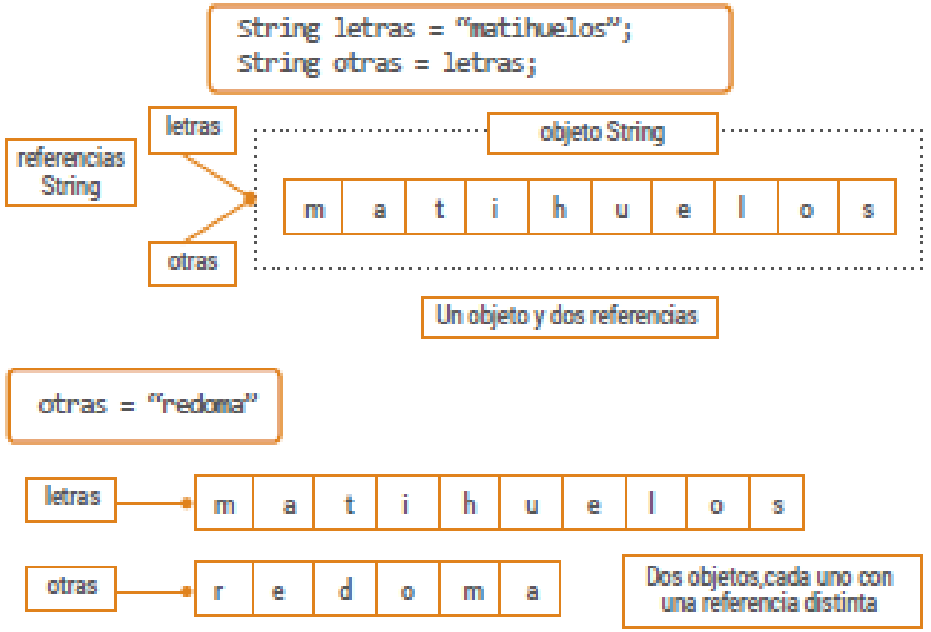
Cadenas

- Cadenas.
  - Las cadenas se pueden considerar como secuencias de caracteres almacenados en posiciones consecutivas de memoria.
  - Su representación de literales como secuencia de caracteres encerrados entre comillas dobles(“).
  - Las cadenas realmente son objetos instanciados de la clase String.
  - El formato para crear un objeto de tipo String es el siguiente:

```
String <identificador_referencia> = new String(<cadena>);
String <identificador_referencia>= <cadena>;
```

Cadenas

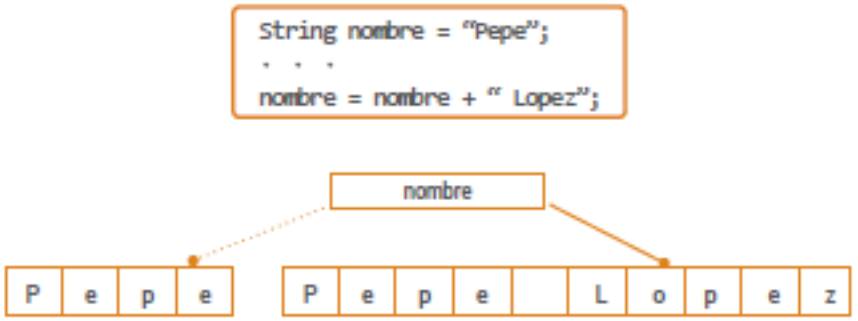
- Relación entre el objeto String y el identificador_referencia.



Virrey Morcillo
Instituto de Educación Secundaria

Cadenas

- La característica más importante de la clase String es que es inmutable no podemos cambiar su contenido.
- Al asignar una valor diferente a una referencia de cadena, estamos creando un objeto cadena nuevo con un nuevo contenido.



Virrey Morcillo
Instituto de Educación Secundaria

Cadenas

- Al ser las cadenas objetos de la clase String, tienen como funcionalidad todos los métodos definidos en String.

Método	Funcionalidad
boolean equals(<cadena>)	Retorna true si la cadena que invoca contiene los mismos caracteres que <cadena>. El operador == no compara los caracteres. compara si las dos referencias referencian al mismo objeto.
int length()	Obtiene la longitud de la cadena
char charAt(<indice>)	Devuelve el carácter cuyo índice en la cadena es <índice>, los índices siempre son enteros.
int compareTo(<cadena>)	Retorna un valor negativo si la cadena que invoca es menor que <cadena>, valor positivo si es mayor que <cadena> y cero si son iguales.
int indexOf(<cadena>)	Busca en la cadena que invoca la subcadena especificada por <cadena>. Devuelve el índice de la primera coincidencia o -1 en caso de encontrarla.
int lastIndexOf(<cadena>)	Busca en la cadena que invoca la subcadena especificada por <cadena>. Devuelve el índice de la última coincidencia o -1 en caso de encontrarla.
String substring(<inicio_índice>)	Devuelve un nuevo String que contiene todos los caracteres del String que invoca desde el índice <inicio_índice> hasta el final.
String substring(<inicio_índice>, <fin_índice>)	Devuelve un nuevo String que contiene todos los caracteres del String que invoca desde el índice <inicio_índice> hasta <fin índice>, sin incluir este.
String toLowerCase()	Devuelve un nuevo String que tiene todos los caracteres del que invoca pero en minúsculas.
String toUpperCase()	Devuelve un nuevo String que tiene todos los caracteres del que invoca pero en mayúsculas.
String trim()	Devuelve un nuevo String que igual que el invocado pero habiendo eliminado todos los espacios en blanco.
String valueOf(<valor_de_un_tipo>)	Devuelve un nuevo String resultado de convertir a cadena el valor que recibe como parámetro. Esta función es static, por tanto no se invoca con ningún objeto sino con el nombre de la clase String: String.valueOf(1_345_231) //retorna "1345231"
String replace(<caracter1>, <caracter2>)	Devuelve un nuevo String en el que se han remplazado las apariciones de <caracter1> por <caracter2>.

Virrey Morcillo
Instituto de Educación Secundaria

Arrays Multidimensionales

- Los arrays multidimensionales utilizan más de un índice para acceder a sus elementos.
- La estructura más usada es la de dos dimensiones → tabla.
- Creación de una tabla:
  - 1º Opción:
    - <tipo> identificador_tabla[][] = new <tipo>[num_filas][num_columnas].
  - 2º Opción:
    - <tipo> identificador_tabla>[][];
    - identificador_tabla = new <tipo> [num_filas][num_columnas];
  - 3º Opcion:
    - <tipo>identificador_tabla[][] = new <tipo>[num_filas][];
    - identificador_tabla[fila_]=new <tipo>[num_columnas];
  -

Virrey Morcillo
Instituto de Educación Secundaria

Arrays Multidimensionales

- Crear una tabla con 10 filas y 6 columnas. En cada uno de sus casillas almacenar un posible resultado de futbol (1, X, 2). El resultado debe ser aleatorio.
- Una vez llenada la tabla, mostrar el resultado de la misma en formato tabla:
  - 1º Fila: 1 X 2 X 2 1
  - 2º Fila: 2 X 1 X 2 2
  - ……
  - ……
  - 10º Fila: 1 2 X 2 1 2

Virrey Morcillo
Instituto de Educación Secundaria

Enumeraciones

- Una enumeración es una lista de constantes con un nombre que define un tipo nuevo.
- Una variable de tipo enumeración solo puede almacenar uno de los valores de la lista.
- Para crear una enumeración utilizamos la palabra reservada num.

```
enum <identificador> { <lista_cosntantes> }
```

```
enum EstadoCivil {
      SOLTERO, CASADO, VIUDO, SEPARADO, DIVORCIADO
}
EstadoCivil ec = CASADO;
if (ec == EstadoCivil.SOLTERO){    /*sentencias*/}
switch( ec ){
      case SOLTERO:
            //...
      case CASADO:
            //...
      }
```

Virrey Morcillo
Instituto de Educación Secundaria

Enumeraciones

- En Java es convención estilística poner las constantes en mayúsculas.
- Todas las enumeraciones cuentan con métodos predefinidos:
  - **values():** devuelve un array que contiene la lista de las constantes de la enumeración.
  - **valuesOf(String cadena),** devuelve la constante de enumeración que se corresponde con la cadena pasada como parámetro.
  - **ordinal(),** devuelve el número de la posición de la constante en la lista.
- Las enumeraciones se crean en el ámbito de la clase, no se pueden crear dentro del ámbito static de la función main.

Virrey Morcillo
Instituto de Educación Secundaria

Enumeraciones

```
            Random aleatorio = new Random();
            int num = aleatorio.nextInt(9)+1;

            int posicion=0;
            EstadoCivil [] valores_enumeracion = EstadoCivil.values();
            for(EstadoCivil estado: valores_enumeracion){
                  posicion = estado.ordinal();
                  System.out.println("La constante número "+ posicion+
                              " es "+estado.toString());

            }

            EstadoCivil [] estados = new EstadoCivil[num];
            int nestados = EstadoCivil.values().length;
            for (int i=0; i<num; i++ ){
                  estados[i]=valores_enumeracion[aleatorio.
nextInt(nestados)];
            }

            posicion=0;
            for(EstadoCivil estado: estados){
                  System.out.println("estados["+posicion+"]:
"+estado);
                  posicion++;

            }
```