Benemérita Universidad Autónoma de Puebla

Práctica #5. POO en PHP

Aplicaciones web

Facultad de Ciencias de la Computación

Equipo:

Vargas Arenas Pedro - 201734553

Descripción

En este documento se describe el desarrollo de la quinta práctica de la materia que consiste en elaborar un programa que realice y muestre el resultado de las operaciones unión, intersección y diferencia de dos conjuntos utilizando la programación orientada a objetos. El usuario ingresará el tamaño de los dos conjuntos en un formulario, los datos de dichos conjuntos se generarán aleatoriamente.

Desarrollo

Formulario

El usuario debe ingresar el tamaño de los conjuntos a operar, para ello se desarrolló un pequeño formulario en el archivo *index.php* que contiene dos campos de texto y un botón. La información ingresada en dichos campos de texto será enviada al *principal.php* donde se encuentran los objetos y llamados a las funciones para ejecutar las operaciones.

```
<form action="principal.php" method="POST" style="text-align:center;">
      <label>Introduzca el tamaño del primer conjunto:</label>
      <input type="text" name="tam1">
      <br>
      <label>Introduzca el tamaño del segundo conjunto:</label>
      <input type="text" name="tam2">
      <br>
      <p><input type="submit" value="Enviar datos"></p>
```

Operaciones de los conjuntos

En el archivo *conjunto.php* se encuentra la clase *Conjunto* que contiene todas las operaciones antes mencionadas y se inicializan los conjuntos. La clase cuenta con dos atributos que son *$tam* para almacenar el tamaño del conjunto, el valor es asignado dentro del constructor y *$con* donde se guarda el conjunto creado. Para iniciarlizar el conjunto se utiliza la función *iniciarConjunto()*, aquí se crea la cantidad de valores deseados dentro del rango de 1 a 20.

```
public function iniciarConjunto(){
      for ($i=0; $i < $this->tam; $i++)
            $this->con[$i] = rand(1, 20);
}
```

La operación unión se ejecuta en la función *union($conjunto1, $conjunto2)* que recibe dos conjuntos. Dentro de la función se recorre el primer conjunto para comparar cada elemento con todo el segundo conjunto al igual que con el conjunto de salida, así se verifica que en el conjunto de salida solo se encuentren los elementos del primer conjunto que no estén en el segundo conjunto.

```
public function union($conjunto1, $conjunto2){

        $n = 0;

        for ($i = 0; $i < $conjunto1->tam; $i++)
           if(!in_array($conjunto1->con[$i], $conjunto2->con) && !in_array($conjunto1->con[$i],
$this->con)){
              $this->con[$n] = $conjunto1->con[$i];
              $n++;
           }
        $this->comprobarConjunto($n, $conjunto2);
}
```

Para anexar el segundo conjunto, se utiliza la función *comprobarConjunto($n, $conjunto2)*. Aquí se recorre dicho conjunto para verificar que no existan elementos repetidos dentro del mismo. Finalmente, se anexa al conjunto de salida.

```
        private function comprobarConjunto($ , $ ){

                $ = FALSE;

                sort($ );
                for ($  = 0; $  < $ ; $ ++){
                   for ($  = $  + 1; $  < $ ; $ ++)
                      if($ [$ ] == $ [$ ])
                         $existe = TRUE;
                   if(!$ ){
                      $this-> [$ ] = $ [$ ];
                      $ ++;
                   }
                   $ = FALSE;
                }
        }
```

La operación intersección se logra mediante la función *interseccion($conjunto1, $conjunto2)* que recibe dos conjuntos. Dentro de la función se recorre el primer conjunto para comparar cada elemento con todo el segundo conjunto, si se encuentra, quiere decir que existe intersección. De igual manera, se compara con el conjunto de salida para evitar que se repitan elementos.

```
public function interseccion($conjunto1, $conjunto2){

        $n = 0;

        for ($i = 0; $i < $conjunto1->tam; $i++)
           if(in_array($conjunto1->con[$i], $conjunto2->con) && !in_array($conjunto1->con[$i],
$this->con)){
              $this->con[$n] = $conjunto1->con[$i];
              $n++;
           }
}
```

La operación diferencia se realiza a través de la función *diferencia($conjunto1, $conjunto2)* en la cual se recorre el primer conjunto para encontrar los elementos que no estén en el segundo conjunto, estos se almacenan en el conjunto de salida.

```
public function diferencia($conjunto1, $conjunto2){

        $n = 0;

        for ($i = 0; $i < $conjunto1->tam; $i++)
           if(!in_array($conjunto1->con[$i], $conjunto2->con) && !in_array($conjunto1->con[$i],
$this->con)){
               $this->con[$n] = $conjunto1->con[$i];
               $n++;
           }
}
```

Creación de objetos

Dentro del archivo *principal.php* se ejecuta la creación y utilización de los objetos. Primero se importa el archivo *conjunto.php* que contiene la clase. Posteriormente, se recibe el tamaño de los dos conjuntos mediante la propiedad *$_REQUEST[“]*, en la instancia de ambos conjuntos, estos datos son enviados como parámetros al constructor, después se instancian los conjuntos que tendrán la unión, la intersección, la diferencia entre el primer conjunto y el segundo y la diferencia entre el segundo conjunto y el primero.

```
$C1 = new Conjunto($tam1);
$C2 = new Conjunto($tam2);
$union = new Conjunto(0);
$interseccion = new Conjunto(0);
$diferenciaC1C2 = new Conjunto(0);
$diferenciaC2C1 = new Conjunto(0);
```

Después, se inician los dos conjuntos que serán operados y se muestran.

```
$  ->iniciarConjunto();
$  ->iniciarConjunto();
echo "<h3>Conjunto 1</h3>";
$  ->mostrar();
echo "<h3>Conjunto 2</h3>";
$  ->mostrar();
```

Posteriormente, se ejecutan las operaciones pasando como parámetro los conjuntos previamente creados.

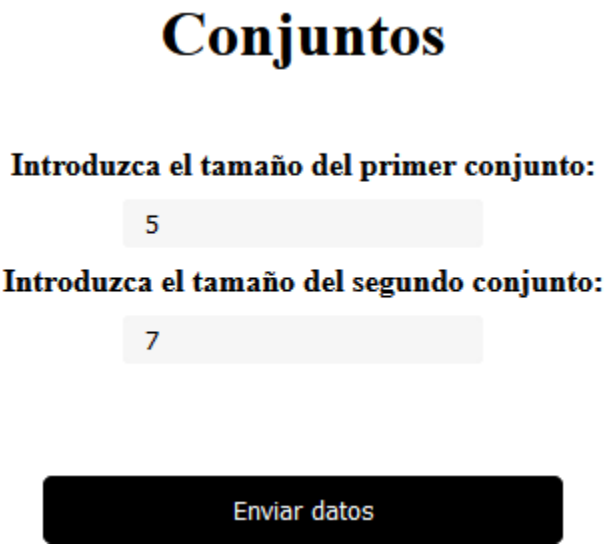
```
$union->union($C1, $C2);
$interseccion->interseccion($C1, $C2);
$diferenciaC1C2->diferencia($C1, $C2);
$diferenciaC2C1->diferencia($C2, $C1);
```

Finalmente, se muestra el resultado de las operaciones.

```
echo "<br><h3>Unión</h3>";
$union->mostrar();
echo "<hr>";
echo "<h3>Intersección</h3>";
$interseccion->mostrar();
echo "<hr>";
echo "<h3>Diferencia de C1 y C2</h3>";
$diferenciaC1C2->mostrar();
echo "<hr>";
echo "<h3>Diferencia de C2 y C1</h3>";
$diferenciaC2C1->mostrar();
```

Ejecución

Primeramente, se debe activar WampServer para activar un servidor y así lograr ejecutar PHP en el navegador. Posteriormente, se ejecuta el archivo index.php que contiene el formulario.



En este caso, se creará un conjunto de cinco elementos y otro con siete elementos. Posteriormente, se da click en “Enviar datos”.

Conjunto 1

6 9 12 18 19

Conjunto 2

2 10 10 12 15 17 19

Elementos de los dos conjuntos

Unión

2 6 9 10 12 15 17 18 19

Intersección

12 19

Diferencia de C1 y C2

6 9 18

Diferencia de C2 y C1

2 10 15 17

Resultados de las operaciones

Referencias

https://developer.mozilla.org/es/docs/Learn/Forms/Your_first_form

https://www.php.net/manual/es/language.oop5.php

https://diego.com.es/instancia-de-clases-en-php

https://diego.com.es/arrays-en-php