Министерство образования и науки Российской Федерации

Ульяновский Технический университет

Кафедра: Вычислительная техника

Дисциплина: Дискретная математика

Отчет по выполненным лабораторным работам по дисциплине

Дискретная математика

Выполнил:

Студент: ИВТАПбд-11

Вершинин Дмитрий Вячеславович

Проверил:

Игонин Андрей Геннадьевич

Ульяновск, 2017

**Лабораторная работа №1.**

**Техническое задание:** Требуется разработать программу на языке PHP, выполняющую объединение, пересечение, дополнение и симметрическую разность двух множеств. Ввод множеств осуществляется пользователем. Каждый элемент множества должен удовлетворять следующим условиям. На первом месте элемента находится цифра, на втором - четная цифра, на третьем - нечетная цифра и на четвертом - цифра. Необходимо выполнить проверку вводимых пользователем данных.

**Ход выполнения:** С помощью функции explode из формы получаем значения элементов обоих множеств, которые будут храниться в двух соответствующих массивах. Для удобства проверку вводимых значений и каждую из операций с множествами удобно вынести в отдельную функцию. В функции проверки необходимо проверить количество цифр в каждом элементе множества и выяснить удовлетворяет ли каждый из этих элементов заданным условиям. Если найдена ошибка, необходимо вывести сообщение пользователю и выделить поле с ошибочным элементом. Также если во введенном множестве встречаются одинаковые элементы, их необходимо удалить.

В функции Объединения множеств записываем в результирующий массив каждый из элементов обоих множеств, затем отбираем из этого массива уникальные элементы, сравнивая элементы данного массива в цикле for и при нахождении повторяющихся элементов смещаем элементы массива, которые находятся правее повторяющегося влево. Таким образом получаем массив, который содержит Объединение двух множеств.

В функции Пересечения множеств с помощью двух вложенных циклов for сравниваем элементы двух множеств и при нахождении одинаковых элементов записываем их в результирующий массив. Таким образом получаем массив, который содержит Пересечение двух множеств.

В функции Дополнения множеств приравниваем результирующий массив одному из множеств. Затем с помощью двух вложенных циклов for сравниваем элементы двух множеств и при нахождении одинаковых элементов удаляем их из результирующего массива.

Симметрическая разность представляет собой массив, состоящий из Дополнения первого множества до второго и второго до первого.

Сначала вызываем функцию Проверки, если проверка пройдена удачно, вызываем поочередно каждую из функций и выводим результат на экран. Если проверка не пройдена выводим соответствующее сообщение об ошибке.

**Вывод**: В данной лабораторной работе была написана программа выполняющую объединение, пересечение, дополнение и симметрическую разность двух множеств.

**Лабораторная работа №2-3**

**Техническое задание:** Требуется разработать программу на языке PHP, определяющую свойства отношения и определяющую является ли данное отношение функцией. Элементы вводятся пользователем в виде матрицы n\*n. Необходимо выполнить проверку ввода.

**Ход выполнения:** С помощью функции explode из формы получаем значения элементов отношения. Для удобство каждое из свойств данного отношения будем определять в отдельной функции.

Проверка введенного отношения будет заключаться в проверке матрицы на бинарность (каждый элемент является 0 или 1), которая выполняется в цикле for с помощью условного оператора. Также необходимо проверить является ли матрица квадратной. Для этого возводим количество элементов в строке матрицы в квадрат и сравниваем с общим количеством элементов, если полученные значения равны, матрица является квадратной, иначе выводим сообщение об ошибке.

В функции проверки на рефлексивность необходимо с помощью цикла for проверить каждый элемент на главной диагонали, если каждый из этих элементов = 1, отношение рефлексивно, иначе не рефлексивно.

В функции проверки на симметричность необходимо сравнить элементы, лежащие выше главной диагонали матрицы с соответствующими элементами лежащими ниже главной диагонали (другими словами проверить совпадает ли матрица с транспонированной). Если элементы совпадают матрица симметрична, если различаются не симметрична.

В функции проверки на транзитивность с помощью трех вложенных циклов for возводим матрицу в квадрат и сравниваем ее с исходной матрицей. Если квадрат матрицы меньше либо равен исходной матрице, то отношение транзитивно.

В функции, которая проверяет является ли отношение функцией с помощью двух вложенных циклов for проверяем сколько единиц содержится в каждой строке матрицы. Для того, что отношение было функцией необходимо, чтобы в каждой строке матрицы стояла ровно одна 1. Если 1 больше или меньше одной отношение не является функцией.

**Вывод:** В данной лабораторной работе была написана программа, определяющая свойства отношения и определяющая является ли данное отношение функцией.

**Лабораторная работа №4.**

**Техническое задание:** Требуется разработать программу на языке PHP, которая будет находить кратчайший путь в графе, а также запоминать и выводить маршрут. Пути между вершинами графа вводятся пользователем в виде матрицы весов.

**Ход выполнения:** Кратчайший путь в графе определяется при помощи алгоритма Дейкстры. С помощью функции explode из формы получаем значения весов ребер графа и записываем их в массив. Получившийся одномерный массив с помощью двух вложенных циклов for переписываем в двумерный. Затем задаем массив итоговых путей (элемент с индексом начального города приравниваем к 0, а остальные к достаточно большому числу, чтобы ребра с таким весом гарантированно не существовало), массив посещенных вершин (в котором будем записывать посещенные вершины) и массив предков (для последующего восстановления маршрута между вершинами).

В начале алгоритма находим не посещённую вершину итоговый путь, в который меньше бесконечности (большого числа). Если такая вершина найдена запоминаем ее индекс. Затем ищем пути из этой вершины во все остальные. Если такие пути существуют, то сравниваем их с теми, которые записаны в массиве итоговых путей. Если новый путь меньше, записываем его в итоговый массив. Затем записываем данную вершину в массив предков и отмечаем ее как посещенную. Повторяем это до тех пор, пока остаются не посещённые вершины.

Затем с помощью массива предков восстанавливаем пути от начальной вершины до остальных и выводим их на экран вместе с минимальными расстояниями.

**Вывод:** В данной лабораторной работе была написана программа нахождения кратчайшего пути в графе.

**Лабораторная работа №5**

**Техническое задание:** Требуется разработать программу на языке PHP, для нахождения матрицы достижимости. Матрица смежности вводится пользователем.

**Ход выполнения:** Матрица достижимости строится с помощью алгоритма Уоршалла. С помощью функции explode из формы получаем значения элементов матрицы. Получившийся одномерный массив с помощью двух вложенных циклов for переписываем в двумерный.

За каждый проход цикла алгоритм генерирует матрицу Wk, используя элементы матрицы Wk-1. Чтобы найти i-ую строку матрицы вычислим выражение Wk-1(i, j) или (Wk-1(i, k) и Wk-1(k, j)) при разных значения j.

Затем с помощью двух вложенных циклов for выводим получившуюся матрицу достижимости на экран.

**Вывод:** В данной лабораторной работе была написана программа нахождения матрицы достижимости.

**Список используемой литературы:**

1. Хаггарти Р. Дискретная математика для программистов. – М.: Техносфера, 2003. - 320с.

**Приложение**

**Лабораторная работа №1**

<html>

<head>

<title>Первая лабораторная работа</title>

</head>

<body>

<H1>Первая лабораторная работа</H1>

<p>Программа выполняет объединение, пересечение, дополнение и симметрическую разность двух множеств. Каждый элемент множества должен удовлетворять следующим условиям. Первый элемент множества цифра, второй - четная цифра, третий - нечетная цифра и четвертый - цифра. Элементы вводить через одинарный пробел.</p>

<form method="get" action="">

<H4>Первое поле</H4>

<input name="massA" class="str1" type="text" placeholder="Множество 1" value="<?=($\_GET['massA'])?>" required>

<br>

<H4>Второе поле</H4>

<input name="massB" class="str2" type="text" placeholder="Множество 2" value="<?=($\_GET['massB'])?>" required>

<br>

<br>

<input type="submit" value="Выполнить операцию">

</form>

<?php

$massA\_str = $\_GET['massA'];

$massA\_num = explode(" ", preg\_replace('/[\s]{2,}/', ' ', $massA\_str));

echo "<br>";

$massB\_str = $\_GET['massB'];

$massB\_num = explode(" ", preg\_replace('/[\s]{2,}/', ' ', $massB\_str));

echo "<br>";

function Validation1($massA){

$n=count($massA);

for ($i = 0;$i < $n;$i++) {

$col=0;

$elem=$massA[$i];

while($elem>0){

$elem=floor($elem/10);

$col++;

}

if($col!=4){

echo '<font color="red"> Каждый элемент множества должен состоять из 4 цифр</font>';

return false;

}

}

for ($i = 0;$i < $n;$i++) {

for ($j = $i+1;$j < $n;$j++) {

if($massA[$i]==$massA[$j]){

for ($k = $j;$k < $n;$k++) {

$massA[$k]=$massA[$k+1];

}

$n--;

$i=-1;

}

}

}

for ($i = 0;$i < $n;$i++){

if(is\_numeric(($massA[$i])/1000)==true and (($massA[$i]/100)%10)%2==0 and (($massA[$i]/10)%10)%2==1 and is\_numeric($massA[$i]%10)==true){

$finmas[$i]=$massA[$i];

}else {

echo '<font color="red"> Один из элементов массива не удовлетворяет условию задачи</font>';

return false;

}

}

return $finmas;

}

function Union($massA,$massB){

$n1=count($massA);

$n2=count($massB);

$sum=$n1+$n2;

for ($i = 0;$i < $n1;$i++) {

$result[$i] = $massA[$i];

}

$k=0;

for ($j = $n1;$j < $sum;$j++) {

$result[$j] = $massB[$k];

$k++;

}

$z=0;

$n=count($result);

for ($i = 0;$i < $n;$i++) {

for ($j = $i+1;$j < $n;$j++) {

if($result[$i]==$result[$j]){

for ($k = $j;$k < $n-1;$k++) {

$result[$k]=$result[$k+1];

}

$z++;

$n--;

}

}

}

for ($i = 0;$i < $sum-$z;$i++) {

echo $result[$i]," ";

}

return true;

}

function Intersection($massA,$massB){

$n1=count($massA);

$n2=count($massB);

$sum=$n1+$n2;

for ($i = 0;$i < $n1;$i++) {

for ($j = 0;$j < $n2;$j++) {

if($massA[$i]==$massB[$j]){

$result[$i]=$massA[$i];

}

}

}

for ($i = 0;$i < $sum;$i++) {

echo $result[$i]," ";

}

return true;

}

function Addition($massA, $massB){

$n1 = count($massA);

$n2 = count($massB);

$sum=$n1+$n2;

for ($i = 0;$i < $n1;$i++) {

$result[$i]=$massA[$i];

}

for ($i = 0;$i < $n1;$i++) {

for ($j = 0;$j < $n2;$j++) {

if($result[$i]==$massB[$j]){

$result[$i]='';

}

}

}

for ($i = 0;$i < $sum;$i++) {

echo $result[$i]," ";

}

}

if(Validation1($massA\_num)==false){

echo '<style> INPUT.str1{border: 4px solid #F00;}</style>';

}else{

echo '<style> INPUT.str1{border: 4px solid #0F0;}</style>';

if(Validation1($massB\_num)==false){

echo '<style> INPUT.str2{border: 4px solid #F00;}</style>';

}else{

echo '<style> INPUT.str2{border: 4px solid #0F0;}</style>';

$massivA=Validation1($massA\_num);

$massivB=Validation1($massB\_num);

$n1 = count($massivA);

$n2 = count($massivB);

$sum=$n1+$n2;

echo "Множество А: ";

for ($i = 0;$i < $n1;$i++) {

echo $massivA[$i]," ";

}

echo "<br>";

echo "Множество B: ";

for ($i = 0;$i < $n2;$i++) {

echo $massivB[$i]," ";

}

echo "<br>";

echo "Объединение множеств: ";

Union($massivA,$massivB);

echo "<br>";

echo "Пересечение множеств: ";

Intersection($massivA,$massivB);

echo "<br>";

echo "Дополнение множеств (A/B): ";

Addition($massivA,$massivB);

echo "<br>";

echo "Дополнение множеств (B/A): ";

Addition($massivB,$massivA);

echo "<br>";

echo "Симметрическая разность множеств: ", Addition($massivA,$massivB)." ".Addition($massivB,$massivA);

}

}

?>

</body>

</html>

**Лабораторная работа №2-3**

<html>

<head>

<title>Вторая лабораторная работа</title>

</head>

<body>

<H1>Вторая - третья лабораторная работа</H1>

<p>Программа определяет свойства отношения заданного в виде бинарной матрицы размерностью nxn</p>

<form method="get" action="">

<H4>Введите матрицу размерностью nxn</H4>

<textarea name="matrix" class="str2" type="text" placeholder="Матрица" required><?=($\_GET['matrix'])?></textarea>

<br>

<br>

<input type="submit" value="Выполнить операцию">

</form>

<?php

$arrStr = $\_GET['matrix'];

$arr = explode(" ", preg\_replace('/[\s]{2,}/', ' ', $arrStr));

$size = count($arr);

$strSize = sqrt($size);

function Reflexivity($array, $n, $strSize){

for($i = 0; $i < $n; $i = $i + $strSize + 1){

if ($array[$i]==0){

echo "Отношение не рефлексивно <br>";

return 0;

}

}

echo "Отношение рефлексивно <br>";

}

function Symmetry($array, $strSize){

$diag=0;

for($k = 1;$k < $strSize;$k++){

$counter=1;

for($i = $diag+1; $i < ($strSize\*$k); $i++){

if($array[$i]!=$array[$diag+($strSize\*$counter)]){

echo "Отношение не симметрично <br>";

return 0;

}

$counter++;

}

$diag = $diag + $strSize + 1;

}

echo "Отношение симметрично <br>";

}

function Transitivity($array, $strSize){

$k=0;

for($i=0;$i<$strSize;$i++){

for($j=0;$j<$strSize;$j++){

$arr2[$i][$j]=$array[$k];

$k++;

}

}

for ($i=0;$i< $strSize;$i++){

for($j=0;$j<$strSize;$j++){

$result[$i][$j]=0;

for($k=0;$k<$size;$k++){

$result[$i][$j]+=$array[$i][$k]\*$array[$k][$j];

}

}

}

$begin=0;

$counter=0;

$step=$strSize;

$size=$strSize\*$strSize;

for($l=0;$l<$size;$l++){

$result[$l]=0;

}

for($j=0;$j<$size;$j++){

for($i=$begin; $i<$begin+$strSize; $i++){

$result[$counter]+=$array[$i]\*$array[($step+$begin)];

if($result[$counter]>0){

$result[$counter]=1;

}

$step+=$strSize;

}

$counter++;

if((($counter % $strSize)==0) && $counter!=0){

$begin+=$strSize;

}

}

for($z=0;$z<$size;$z++){

if($array[$z] < $result[$z]){

echo "Отношение не транзитивно <br>";

return 0;

}

}

echo "Отношение транзитивно <br>";

}

function Lab3($array, $strSize){

$counter=0;

$begin=0;

for($j=0;$j<$strSize;$j++){

for($i=$begin;$i<($begin+$strSize);$i++){

if($array[$i]==1){

$counter++;

}

}

if($counter != 1){

echo"Данное отношение не является функцией<br>";

return 0;

}

$counter=0;

$begin+=$strSize;

}

echo "Данное отношение является функцией";

}

for($i=0;$i<$size;$i++){

if(($arr[$i]>1) || ($arr[$i]<0)){

echo"Матрица должна быть бинарной, один из элементов матрицы не удовлетворяет условию";

return 0;

}

}

if(round($strSize)\*round($strSize) == $size){

Reflexivity($arr, $size, $strSize);

Symmetry($arr, $strSize);

Transitivity($arr, $strSize);

Lab3($arr, $strSize);

} else{

echo "Матрица не квадратная, введите матрицу размерностью nxn";

}

?>

</body>

</html>

**Лабораторная работа №4**

<html>

<head>

<title>Четвертая лабораторная работа</title>

</head>

<body>

<H1>Четвертая лабораторная работа</H1>

<p>Программа определяет кратчайший маршрут между городами</p>

<form method="get" action="">

<H4>Введите матрицу размерностью nxn</H4>

<textarea name="matrix" class="str2" type="text" placeholder="Матрица" required><?=($\_GET['matrix'])?></textarea>

<H4>Введите начальную точку</H4>

<input name="begin" type="text" placeholder="Начальная точка" value="<?=($\_GET['begin'])?>" required>

<br>

<br>

<input type="submit" value="Выполнить операцию">

</form>

<?php

$arrStr = $\_GET['matrix'];

$arr = explode(" ", preg\_replace('/[\s]{2,}/', ' ', $arrStr));

$begin = ($\_GET['begin']-1);

$size = count($arr);

$strSize = sqrt($size);

$inf=30000;

if($begin+1>$strSize || $begin<0){

echo "Такого города не существует";

return 0;

}

$k=0;

for($i=0;$i<$strSize;$i++){

for($j=0;$j<$strSize;$j++){

$arr2[$i][$j]=$arr[$k];

$k++;

}

}

for($i=0;$i<$strSize;$i++){

$result[$i]=$inf;

$visited[$i]=0;

$prev[$i]=-15;

}

$result[$begin]=0;

do {

$minindex=$inf;

$min=$inf;

for($i=0;$i<$strSize;$i++){

if ($visited[$i]==0 && $result[$i]<$min){

$min=$result[$i];

$minindex=$i;

}

}

if($minindex != $inf){

for($i=0;$i<$strSize;$i++){

if($arr2[$minindex][$i]>0){

if($min+$arr2[$minindex][$i]<$result[$i]){

$result[$i]=$min+$arr2[$minindex][$i];

$prev[$i]=$minindex;

}

}

}

$visited[$minindex]=1;

}

} while ($minindex<$inf);

print\_r($prev);

echo "Длина пути из города ";

echo $begin+1 . " до остальных.";

echo "<br><br>";

for($z=0;$z<$strSize;$z++){

if($result[$z]!=$inf){

echo "Расстояние из города ";

echo $begin+1 ." до города ";

echo $z+1 ." = ";

echo $result[$z];

echo ".......Маршрут ";

$finalpath = array();

$prev[$begin]=$begin;

$k=0;

$index=$z;

echo $begin+1;

echo " -> ";

while($prev[$index]!= $begin){

$finalpath[$k]=$prev[$index]+1;

$index=$prev[$index];

$k++;

}

for($j=count($finalpath)-1;$j>=0;$j--){

echo $finalpath[$j];

echo " -> ";

}

echo $z+1;

echo "<br>";

}

}

?>

</body>

</html>

**Лабораторная работа №5**

<html>

<head>

<title>Пятая лабораторная работа</title>

</head>

<body>

<H1>Пятая лабораторная работа</H1>

<p>Программа вычисляет матрицу достижимости</p>

<form method="get" action="">

<H4>Введите матрицу размерностью nxn</H4>

<textarea name="matrix" class="str2" type="text" placeholder="Матрица" required><?=($\_GET['matrix'])?></textarea>

<br>

<br>

<input type="submit" value="Выполнить операцию">

</form>

<?php

$arrStr = $\_GET['matrix'];

$arr = explode(" ", preg\_replace('/[\s]{2,}/', ' ', $arrStr));

$size = count($arr);

$strSize = sqrt($size);

function Warshall($array, $strSize){

$k=0;

for($i=0;$i<$strSize;$i++){

for($j=0;$j<$strSize;$j++){

$arr2[$i][$j]=$array[$k];

$result[$i][$j]=$array[$k];

$k++;

}

}

for($k=0;$k<$strSize;$k++){

for($i=0;$i<$strSize;$i++){

for($j=0;$j<$strSize;$j++){

$result[$i][$j]=$result[$i][$j] | ($result[$i][$k] & $result[$k][$j]);

}

}

}

for($i=0;$i<$strSize;$i++){

for($j=0;$j<$strSize;$j++){

echo $result[$i][$j];

echo " ";

}

echo "<br>";

}

}

Warshall($arr, $strSize);

?>

</body>

</html>