GET - метод для отправки данных от клиента на сервер.

__DIR__ - магическая константа, получает путь (директория файла).

Require – выражение для включения и выполнения указанного файла (в случае возникновения ошибки выдаст фатальную ошибку).

Action – атрибут form (формы) в html, указываем адрес программы или документа, который обрабатывает данные формы (action="/result.php").

Input – входит в тег form, создает ячейку для ввода информации пользователем. Необходимо указать атрибут type="text" и задать имя name="x1">.

Submit – значение атрибута type тега Input. Позволяет отправить данные, которые ввёл пользователь, на сервер (type="submit"). Название на кнопке задается атрибутом value="Посчитать".

Select – тег HTML позволяет создать элемент интерфейса в виде раскрывающегося списка, а также список с одним или множественным выбором. Есть атрибут name (name="operation").

Option – атрибут тега Select, с помощью которого создаются пункты. Значение пунктов задается с помощью атрибута value (value="+").

null coalesce operator – ////{в php есть обычный тернарный оператор:

echo $count ? $count : 10; // выводит 10

также есть сокращенный вариант, когда можно пропустить второй элемент, если он равен первому:

echo $count ?: 10;

null coalesce operator - возвратится первое существующее не null значение (оператор объединения с null (??)):

<?php

// Извлекаем значение $_GET['user'], а если оно не задано,то возвращаем 'nobody'

$username = $_GET['user'] ?? 'nobody';

// Это идентично следующему коду:

$username = isset($_GET['user']) ? $_GET['user'] : 'nobody';

}////

Пишем калькулятор

1. Сначала создаем файл index.php и в нём создаем форму HTML

```
<html>
    <head>
        <title>Калькулятор</title>
    </head>
    <body>
        <form action="/result.php" method="get">
            <input type="text" name="x1">
            <select name="operation">
                <option value="+">+</option>
```

```
                <option value="-">-</option>
                <option value="/">/</option>
                <option value="*">*</option>
            </select>
            <input type="text" name="x2">
            <input type="submit" value="Посчитать">
        </form>
    </body>
</html>
```

2. Создаем файл result.php

```php
<?php

$result =  require __DIR__ .'/calc.php';

?>

<html>
    <head>
        <meta charset="UTF-8">
        <title>Калькулятор</title>
    </head>
    <body>
        <b>Результат вычислений:</b>
        <br>
        <?= $result ?>
    </body>
</html>
```

2.1 Открываем тег PHP и создаем переменную $result, в которую помещаем выражение require

2.2 Выражение require с помощью константы __DIR__ находит директорию, в которой находится файл result.php.

2.3 Найденный путь директории с помощью конкатенации добавляем к имени файла calc.php, которые будет создан далее. ($result = require __DIR__ .'/calc.php';). Таким образом получили переменную $result, значением которой является – подключение файла calc.php. Закрываем тег PHP.

2.4 Далее в HTML в теге <body> пишем текст “Результат вычислений:”, а ниже открываем тег PHP и помещаем вывод (<?=) переменной $result.

3. Создаем файл calc.php

```php
<?php

if(empty($_GET)){
    return 'Ничего не передано';
}

if(empty($_GET['operation'])){
    return 'Операция не передана';
}

$x1 = $_GET['x1'] ?? null;
$x2 = $_GET['x2'] ?? null;

if ($x1 === null || $x2 === null) {
    return 'Аргументы 1 или 2 не переданы';
}

$operation = $_GET['operation'];

if (!is_numeric($x1) || !is_numeric($x2)) {
    return 'Введите число';
}

$x1 = (float)$x1;
$x2 = (float)$x2;

if (is_numeric($x1) && is_numeric($x2)){
    switch ($operation){
        case '+':
            $result = $x1 + $x2;
            break;
        case '-':
            $result = $x1 - $x2;
            break;
        case '/':
            $result = $x2 != 0 ? ($x1 / $x2) : 'На ноль делить нельзя';
            break;
        case '*':
            $result = $x1 * $x2;
            break;
        default:
            return 'Операция не поддерживается';
    }
}else {
    return 'Введите число';
}
$expression =  $x1 . ' ' . $operation . ' ' . $x2 . ' = ';

return $expression . $result;
```

3.1 Проверяем есть ли что-то в переданном массиве $_GET (т.е. введены ли пользователем какие-то данные в ячейки формы). Проверяем с помощью функции empty. Если массив $_GET пустой, то empty($_GET) вернёт true и с помощью return выведем 'Ничего не передано'.

3.1 Проверяем выбрана ли операция в массиве $_GET (помним, что в файле index.php мы создали два input (input type="text" name="x1" и input type="text" name="x2"), в которых пользователь вводит аргументы для вычисления): с помощью тега Select (select name="operation")и тега option, в котором есть атрибут value со значением “+”, “-”, “*” или “/” пользователь должен был выбрать операцию. Значение value присваивается переменной operation. Если operation пуста, а это мы проверяем с помощью функции empty (empty($_GET['operation']), то с помощью return выведем 'Операция не передана''.

3.2 Чтобы калькулятор вёл себя корректно, когда пользователь не вводит аргументы, необходимо сделать следующее:

3.2.1 Помещаем в переменную x1 элемент массива get с таким же названием

$x1 = $_GET['x1'] ?? null;

3.2.2 С помощью оператора объединения со значением NULL (null coalesce operator) проверяем, если $_GET['x1'] не задан, то в $x1 записываем null

3.2.3 Далее проверяем, если $x1 === null или $x2 === null, то с помощью return выведем Аргументы 1 или 2 не переданы''.

3.3 Создаем переменную $operation, в которую помещаем значение эелемента operation в массиве get.

3.4 Проводим проверку, являются ли введенные пользователем аргументы, числами. Проверяем с помощью функции is_numeric. Если is_numeric($x1) не является true или is_numeric($x2) не является true, то с помощью return выведем “Введите число''.

3.5 Далее для корректного деления аргументов приводим введенные пользователем числа к дробному типу $x1 = (float)$x1;.

3.6 Теперь проверяем действительно ли $x1 и $x2 являются числами, если нет, то будет выведено return 'Введите число', а если да, то:

С помощью оператора switch сравниваем, что ввёл пользователь в переменную $operation (“+”, “-”, “*” или “/”) и выполняем соответствующую операцию с аргументами. Результат операции заносим в переменную $result. Ставим в default - <return 'Операция не поддерживается';>, если вдруг вычисление окажется невозможным.

3.7. Создаем строковое выражение для красивого отображения результата и помещаем в переменную $expression = $x1 . ' ' . $operation . ' ' . $x2 . ' = ';.

3.8. С помощью конкатенации склеиваем выражение $expression с $result.

3.9. Последней строчкой возвращаем результат, который уходит на страницу result.php, где и отобразится для пользователя.