NAMA : Muhammad Rafli Pradipta

NIM : 225150601111005

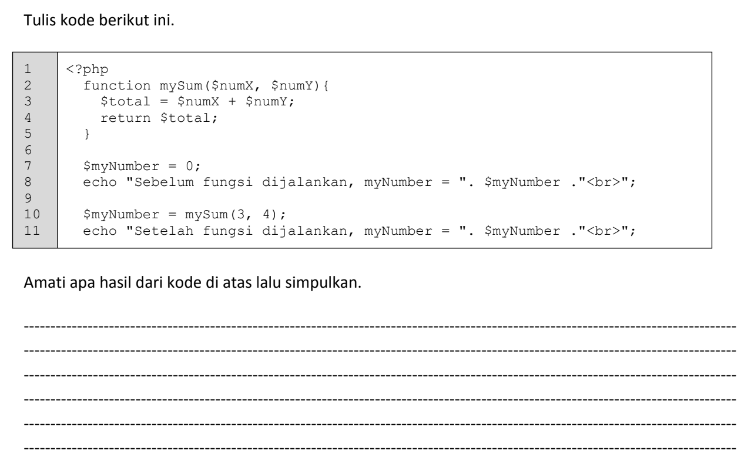
KELAS : PWTI-1

BAB : 4-Fungsi

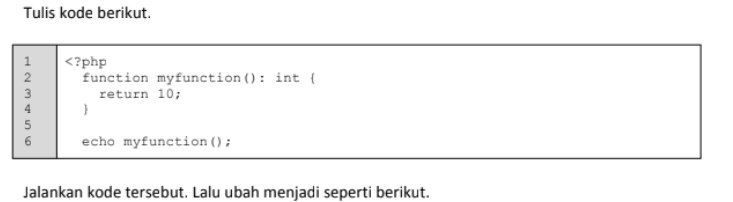
ASISTEN : 1. Agustinus Parasian Tampubolon

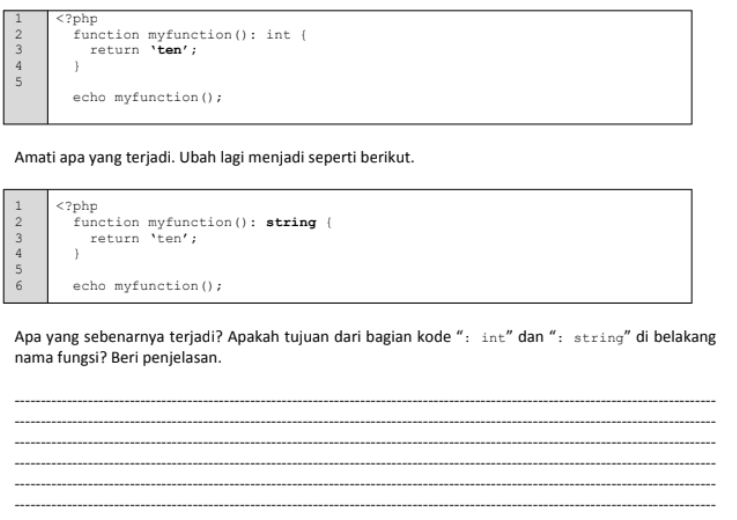
2. Muhammad Dhzuhri Agistian

1. **Langkah-Langkah Praktikum**
2. **Fungsi**

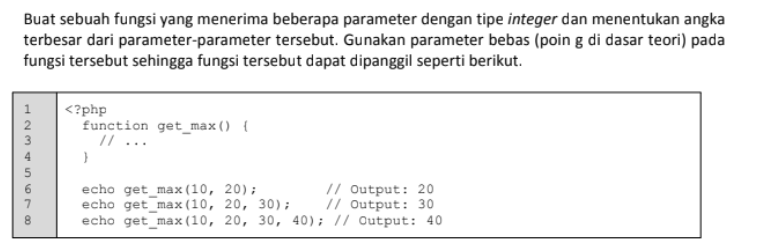


1. Return Value

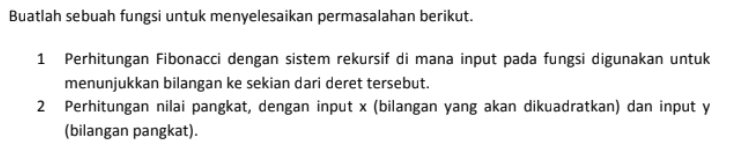




1. Parameter Bebas



1. Rekursi

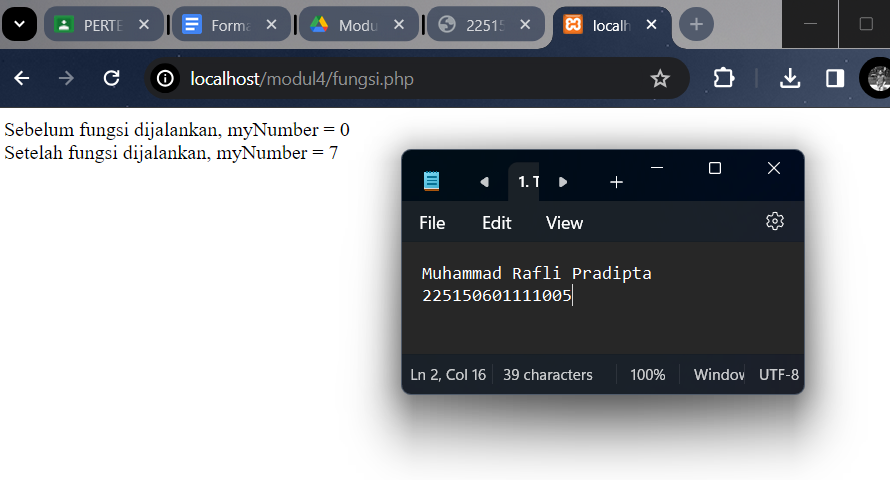


1. **Pembahasan Tugas**
2. Fungsi
   * + 1. **Source code**

**//**Jika ada menggunakan sintaks

|  |
| --- |
| <?php  function mySum($numX, $numY){  $total = $numX + $numY;  return $total;  }  $myNumber = 0;  echo "Sebelum fungsi dijalankan, myNumber = ". $myNumber ."<br>";  $myNumber = mySum(3, 4);  echo "Setelah fungsi dijalankan, myNumber = ". $myNumber ."<br>"; |

* + - 1. **Screenshot**

****

Kode PHP di atas memulai dengan menginisialisasi variabel `$myNumber` dengan nilai 0, kemudian mencetak nilai tersebut. Selanjutnya, fungsi `mySum()` dipanggil dengan argumen 3 dan 4, yang menjumlahkan kedua angka tersebut dan mengembalikan hasilnya. Hasil penjumlahan, yaitu 7, disimpan kembali dalam variabel `$myNumber`, yang kemudian dicetak. Jadi, kode tersebut menunjukkan bagaimana sebuah fungsi dapat dipanggil, mengembalikan nilai, dan memodifikasi nilai variabel di luar ruang lingkup fungsi.

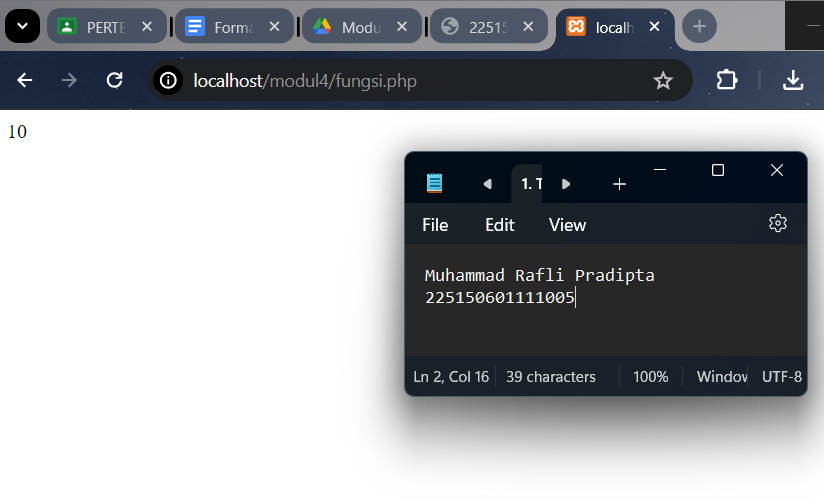
1. **Return Value**

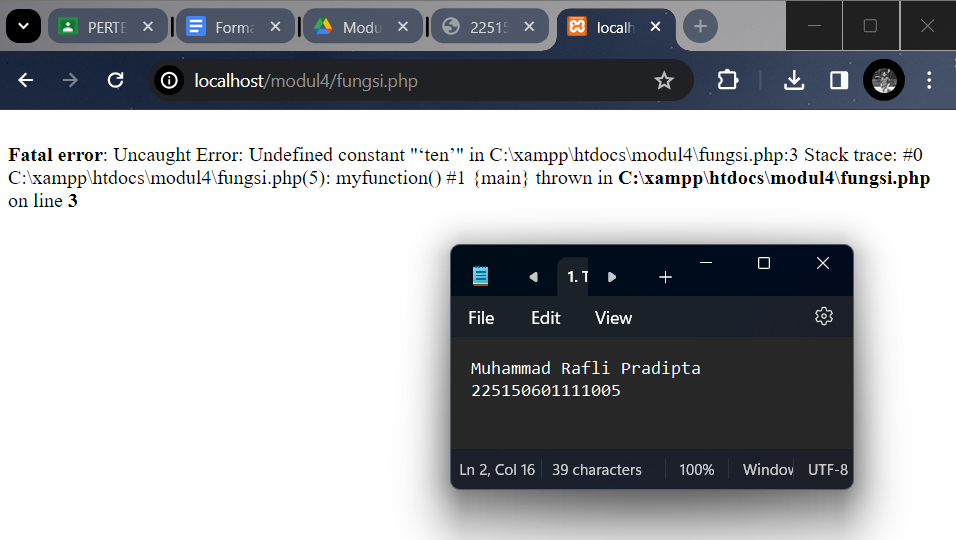
**Source Code**

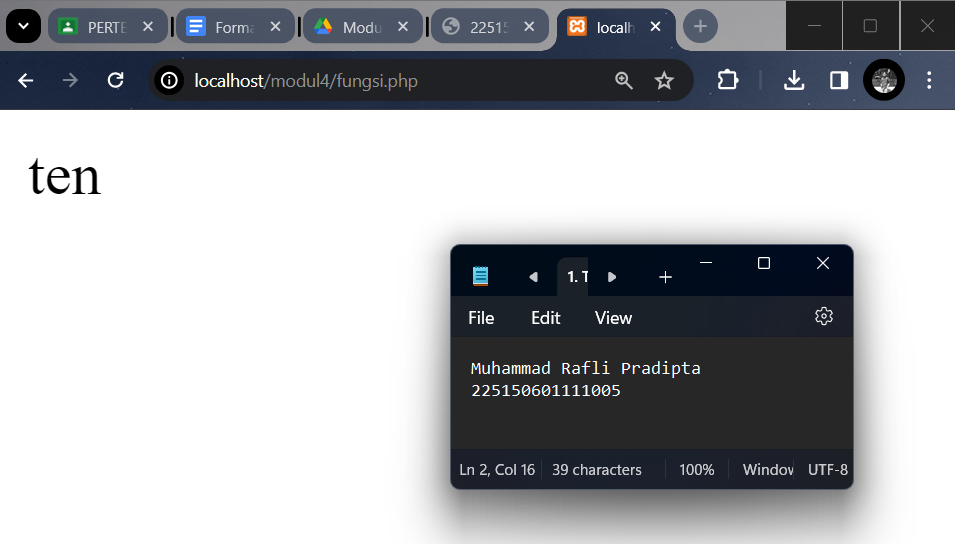
**//**Jika ada menggunakan sintaks

|  |
| --- |
| <?php  function myfunction(): int {  return 10;  }  echo myfunction(); |
| <?php  function myfunction(): int {  return ‘ten’;  }  echo myfunction(); |
| <?php  function myfunction(): string {  return ‘ten’;  }  echo myfunction(); |

**2. Screenshot**





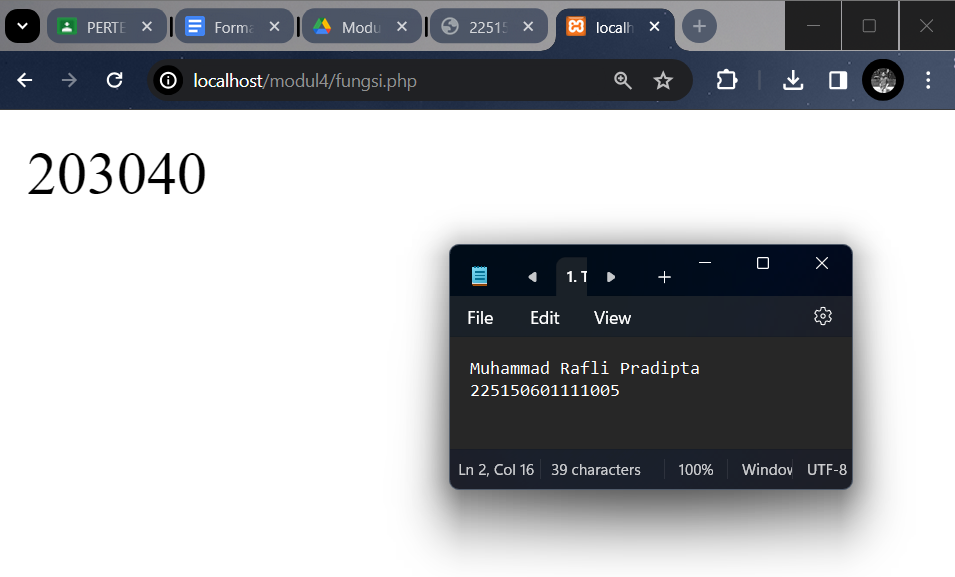


Tujuan penggunaan ": int" atau ": string" di belakang deklarasi nama fungsi ialah untuk mendefinisikan tipe data yang diharapkan akan dikembalikan oleh fungsi tersebut. Dengan menentukan tipe data yang diharapkan, kita bisa mencegah terjadinya error jika nilai yang sebenarnya dikembalikan fungsi tidak sesuai dengan tipenya. Fungsi akan berjalan lancar dan menampilkan hasil sesuai ekspektasi jika nilai yang dikembalikan sesuai dengan tipe data yang telah didefinisikan di deklarasi fungsinya. Penggunaan type hinting ini membantu mencegah terjadinya bug karena ketidaksesuaian antara tipe data yang diharapkan dengan yang dikembalikan oleh suatu fungsi.

1. Parameter Bebas
2. Source Code

|  |
| --- |
| 1. <?php 2. function get\_max(...$numbers) { 3. $max = $numbers[0]; 4. foreach ($numbers as $number) { 5. if ($number > $max) { 6. $max = $number; 7. } 8. } 9. return $max; 10. } 11. echo get\_max(10, 20); *// Output: 20* 12. echo get\_max(10, 20, 30); *// Output: 30* 13. echo get\_max(10, 20, 30, 40); *// Output: 40* |

2. Screenshot



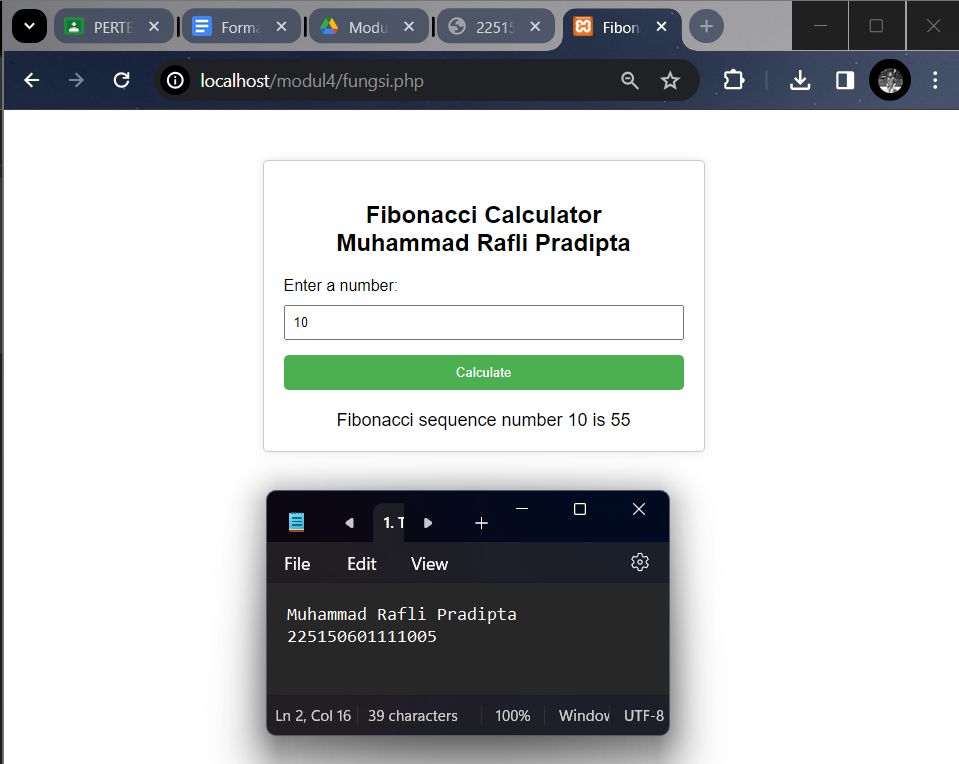
Fungsi get\_max menerima beberapa parameter integer dan menentukan angka terbesar di antara parameter tersebut. Fungsi ini menggunakan parameter bebas, sehingga jumlah parameternya tidak ditentukan. Contohnya, get\_max(10, 20) akan menghasilkan 20, dan get\_max(10, 20, 30) menghasilkan 30. Fungsi ini menggunakan loop foreach untuk iterasi melalui semua parameter dan menemukan nilai terbesar.

1. **Rekursi**
2. **Source Code**

**Perhitungan Fibonacci**

|  |
| --- |
| <?php  function fibonacci($n){      if($n <= 1){          return $n;      } else {          return (fibonacci($n-1) + fibonacci($n-2));      }  }  *// Mengecek apakah ada input dari form*  if(isset($\_POST['submit'])){      $n = $\_POST['number'];      $result = fibonacci($n);  }  ?>  <!DOCTYPE *html*>  <html *lang*="en">  <head>  <meta *charset*="UTF-8">  <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">  <title>Fibonacci Calculator-Muhammad Rafli Pradipta</title>  <style>      body {          font-family: Arial, sans-serif;      }      .container {          max-width: 400px;          margin: 50px auto;          padding: 20px;          border: 1px solid #ccc;          border-radius: 5px;          box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);      }      h2 {          text-align: center;      }      form {          display: flex;          flex-direction: column;      }      label {          margin-bottom: 10px;      }      input[type="number"] {          padding: 8px;          margin-bottom: 15px;      }      input[type="submit"] {          padding: 10px 20px;          background-color: #4CAF50;          color: white;          border: none;          border-radius: 5px;          cursor: pointer;      }      .result {          margin-top: 20px;          text-align: center;          font-size: 18px;      }  </style>  </head>  <body>      <div *class*="container">          <h2>Fibonacci Calculator              <br>Muhammad Rafli Pradipta</h2>          <form *method*="post" *action*="">              <label *for*="number">Enter a number:</label>              <input *type*="number" *id*="number" *name*="number" *min*="0" *required*>              <input *type*="submit" *name*="submit" *value*="Calculate">          </form>          <?php if(isset($result)): ?>              <div *class*="result">                  Fibonacci sequence number <?php echo $n; ?> is <?php echo $result; ?>              </div>          <?php endif; ?>      </div>  </body>  </html> |

1. **Screenshot**

****

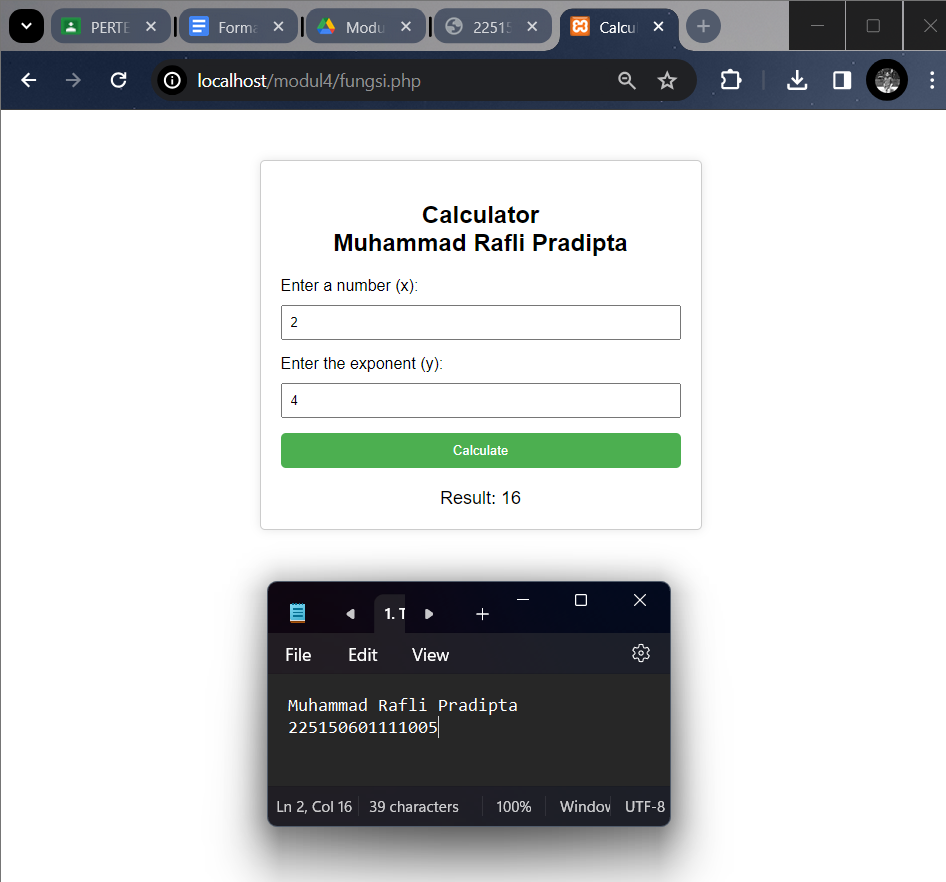
Kode di atas adalah sebuah program PHP yang mengimplementasikan perhitungan deret Fibonacci dengan menggunakan rekursi. Ketika pengguna memasukkan bilangan ke dalam form HTML, nilai tersebut diambil dengan menggunakan metode POST, kemudian diproses oleh PHP. Fungsi `fibonacci()` dijalankan secara rekursif untuk menghitung nilai bilangan yang dimasukkan dalam deret Fibonacci. Hasil perhitungan tersebut kemudian ditampilkan kembali kepada pengguna di halaman HTML dengan menggunakan CSS untuk styling.

1. **Source Code**

**Perhitungan nilai pangkat**

|  |
| --- |
| <?php  function hitungPangkat($x, $y) {      return pow($x, $y);  }  *// Mengecek apakah ada input dari form*  if(isset($\_POST['submit'])){  *// Memeriksa ketersediaan kunci 'x' dan 'y' pada $\_POST*      $x = isset($\_POST['x']) ? $\_POST['x'] : null;      $y = isset($\_POST['y']) ? $\_POST['y'] : null;  *// Memeriksa apakah keduanya tidak null sebelum melakukan perhitungan*      if ($x !== null && $y !== null) {          $result = hitungPangkat($x, $y);      } else {  *// Menetapkan hasil ke null jika salah satu atau kedua input tidak tersedia*          $result = null;      }  }  ?>  <!DOCTYPE *html*>  <html *lang*="en">  <head>  <meta *charset*="UTF-8">  <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">  <title>Calculator</title>  <style>      body {          font-family: Arial, sans-serif;      }      .container {          max-width: 400px;          margin: 50px auto;          padding: 20px;          border: 1px solid #ccc;          border-radius: 5px;          box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);      }      h2 {          text-align: center;      }      form {          display: flex;          flex-direction: column;      }      label {          margin-bottom: 10px;      }      input[type="number"] {          padding: 8px;          margin-bottom: 15px;      }      input[type="submit"] {          padding: 10px 20px;          background-color: #4CAF50;          color: white;          border: none;          border-radius: 5px;          cursor: pointer;      }      .result {          margin-top: 20px;          text-align: center;          font-size: 18px;      }  </style>  </head>  <body>      <div *class*="container">          <h2>Calculator <br> Muhammad Rafli Pradipta</h2>          <form *method*="post" *action*="">              <label *for*="x">Enter a number (x):</label>              <input *type*="number" *id*="x" *name*="x" *required*>              <label *for*="y">Enter the exponent (y):</label>              <input *type*="number" *id*="y" *name*="y" *required*>              <input *type*="submit" *name*="submit" *value*="Calculate">          </form>          <?php if(isset($result)): ?>              <div *class*="result">                  Result: <?php echo $result; ?>              </div>          <?php endif; ?>      </div>  </body>  </html> |

1. **Screenshot**

****

Kode di atas adalah sebuah program PHP yang memungkinkan pengguna untuk menghitung nilai pangkat dari sebuah bilangan. Ketika pengguna memasukkan dua angka melalui form HTML, yaitu bilangan yang akan dipangkatkan (x) dan pangkatnya (y), PHP akan memproses input tersebut dan menghitung nilai pangkat menggunakan fungsi `pow()`. Namun, sebelum mengakses nilai dari `$\_POST['x']` dan `$\_POST['y']`, program melakukan pengecekan terlebih dahulu apakah kedua input tersebut sudah tersedia dalam array `$\_POST`. Jika salah satu atau kedua input tidak tersedia, maka hasil perhitungan akan diatur menjadi null. Setelah proses perhitungan selesai, hasilnya akan ditampilkan di bawah form HTML. Ini memastikan bahwa program tidak akan mencoba mengakses input yang tidak ada, menghindari munculnya pesan kesalahan "Undefined array key".