

**10111002 – AHMAD RAMDANI**

**SISTEM INFORMASI 2B**

JOB SHEET   
PEMROGRAMAN BERORIENTASI OBJEK

PERTEMUAN 1

Dosen Pengampu

Sari Azhariyah, M.Pd.T.  
Usep Abdul Rosid, S.T., M.Kom

|  |  |
| --- | --- |
| POLITEKNIK NEGERI SUBANG |  |
| Jurusan : TIK | Waktu : 240 Menit |
| Program Studi : Sistem Informasi | Topik : |
| Mata Kuliah : PBO | Judul : |
| Tanggal : …………………. | Pertemuan ke : 7 |

# Tujuan : Mahasiswa mampu memahami dan menjelaskan konsep dasar dan mengimplementasi serta berlatih penggunaan elemen pada HTML.

# Alat dan Bahan : XAMPP, Web Browser, Text Editor

# Link Github : https://github.com/Ahmad0602/AhmadRamdani\_SI2B\_PBO.git

**Langkah Kerja :**

1. **Siapkan dan jalankan xampp, web browser, dan text editor**
2. **Buat folder praktikum pemrograman web pada folder htdocs**
3. **Lakukan penulisan sintak program pada text editor**
4. **Simpan file hasil penulisan sintak tersebut kedalam folder praktikum**
5. **Jalankan web browser**
6. **Jalankan web pada mode local dengan mengetikan localhost/folder praktikum/file sintak pada web browser**
7. **Output program akan tampil pada web browser**

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Praktik** | **Output Program** | **Sintak Kode Program** |
| 1. | **Latihan Soal 7.1** |  | <?php  class manusia{  //properties (encapsulation)  public $nama="Ardi";  var $kelas = "SI 2";  //method manusia  function tampilkan\_nama(){  return $this->nama;  }  public function tampilkan\_kelas(){  return $this->kelas;  }  }  //instansiasi class manusia  $manusia = new manusia();  //memanggil method tampilkan\_nama dari class manusia  echo "Nama : ".$manusia->tampilkan\_nama()."<br />";  echo "Kelas : ".$manusia->tampilkan\_kelas();  ?> |
| 2. | **Latihan Soal 7.2** |  | <?php  class manusia{      //property (encapsulation)      private $nama="Ardi";      private $kelas = "SI 2";      //method manusia      private function m\_nama(){          return $this->nama;      }      public function tampilkan\_nama(){          return $this->m\_nama();      }      function tampilkan\_kelas(){          return $this->kelas;      }  }  //instansiasi class manusia  $manusia = new manusia();  //memanggil method tampilkan\_nama dari class manusia  echo "Nama : ".$manusia->tampilkan\_nama()."<br />";  echo "Kelas : ".$manusia->tampilkan\_kelas();  ?> |
| 3. | **Latihan Soal 7.3** |  | <?php  // class manusia  class manusia {      // properti dengan protected      protected $nama = "Ardi";      var $kelas = "SI 2";        // method protected      protected function nama() {          return "Nama : " . $this->nama;      }        // method public untuk akses nama      public function tampilkan\_nama() {          return $this->nama();      }        // ubah jadi public agar bisa dipanggil di luar class      public function tampilkan\_kelas() {          return "Kelas : " . $this->kelas;      }  }    // instansiasi class manusia  $manusia = new manusia();    // memanggil method public  echo $manusia->tampilkan\_nama() . "<br />";  echo $manusia->tampilkan\_kelas();  ?> |
| 4. | **Latihan Soal 7.4** |  | <?php  // buat class komputer  class komputer {      // property dengan hak akses berbeda      protected $jenis\_processor = "Intel Core i7-4790 3.60GhZ";      protected $jenis\_RAM = "DDR 4";      public $jenis\_VGA = "PCI Express";        public function tampilkan\_processor() {          return $this->jenis\_processor;      }        public function tampilkan\_jenisprocessor() {          return $this->jenis\_processor;      }        protected function tampilkan\_ram() {          return $this->jenis\_RAM;      }        protected function tampilkan\_vga() {          return $this->jenis\_VGA;      }        public function tampilkan\_vga2() {          return $this->jenis\_VGA;      }  }    // buat class laptop yang mewarisi class komputer  class laptop extends komputer {      public function display\_processor() {          return $this->jenis\_processor;      }        public function display\_processor2() {          return $this->tampilkan\_processor();      }        public function display\_ram() {          return $this->jenis\_RAM;      }        public function display\_ram2() {          return $this->tampilkan\_ram();      }        public function display\_vga() {          return $this->tampilkan\_vga();      }        public function display\_processorkomputer() {          return $this->jenis\_processor;      }  }    // buat objek dari class  $komputer = new komputer();  $laptop = new laptop();    // jalankan method  echo "Line 61 = " . $komputer->tampilkan\_processor() . "<br />";  echo "Line 62 = " . $laptop->display\_processor() . "<br />";  echo "Line 63 = " . $laptop->display\_processor2() . "<br />";  echo "Line 64 = " . $laptop->tampilkan\_jenisprocessor() . "<br />";  echo "Line 65 = " . $laptop->display\_ram() . "<br />";  echo "Line 66 = " . $laptop->display\_vga() . "<br />";  echo "Line 67 = " . $laptop->display\_processorkomputer() . "<br />";  ?> |
| 5. | **Latihan Soal 7.5** |  | <?php  //Class abstrak (abstract class) didefinisikan sebagai class yang tidak bisa diinstansiasi menjadi object.  //Sebuah class abstrak setidaknya memiliki satu method abstrak (abstract method).  //Untuk menggunakan sebuah class abstrak, kita harus membuat class turunan dari class abstrak tersebut.  //Tujuan dari dibuat abstrak class adalah agar setiap class turunan menggunakan method abstrak pada class induk secara fleksibel.  //Jadi pada method abstrak di class induk hanya terdapat signature saja tanpa isi yang konkrit.  abstract class Hewan {      abstract public function Makan();      abstract public function Bergerak();      abstract public function Beranak();  }  class Burung extends Hewan {      public function Makan() {          return "Burung makan biji-bijian<br/>";      }      public function Bergerak() {          return "Burung bergerak dengan berjalan, terbang dan melompat<br/>";      }      public function Beranak() {          return "Burung beranak dengan bertelur<br/>";      }  }  class Kambing extends Hewan {      public function Makan() {          return "Kambing makan rumput<br/>";      }      public function Bergerak() {          return "Kambing bergerak dengan berjalan dan berlari<br/>";      }      public function Beranak() {          return "Kambing beranak dengan melahirkan<br/>";      }  }  $burung = new Burung;  $kambing = new Kambing;  echo "<b>Perilaku Burung :</b><br/>";  echo $burung->Makan();  echo $burung->Bergerak();  echo $burung->Beranak();  echo "<br/>";  echo "<b>Perilaku Kambing :</b><br/>";  echo $kambing->Makan();  echo $kambing->Bergerak();  echo $kambing->Beranak();  ?> |
| 6. | **Latihan 7.6** |  | <?php  Interface Hewan {      // membuat interface : sebuah class definition yang didalam body nya hanya terdiri dari definisi fungsi tanpa ada implementasi      public function Makan();      public function Bergerak();      public function Beranak();  }  class Burung implements Hewan {      public function Makan() {          return "Burung makan biji-bijian<br/>";      }      public function Bergerak() {          return "Burung bergerak dengan berjalan, terbang dan melompat<br/>";      }      public function Beranak() {          return "Burung beranak dengan bertelur<br/>";      }  }  class Kambing implements Hewan {      public function Makan() {          return "Kambing makan rumput<br/>";      }      public function Bergerak() {          return "Kambing bergerak dengan berjalan dan berlari<br/>";      }      public function Beranak() {          return "Kambing beranak dengan melahirkan<br/>";      }  }  $burung = new Burung;  $kambing = new Kambing;  echo "<b>Perilaku Burung :</b><br/>";  echo $burung->Makan();  echo $burung->Bergerak();  echo $burung->Beranak();  echo "<br/>";  echo "<b>Perilaku Kambing :</b><br/>";  echo $kambing->Makan();  echo $kambing->Bergerak();  echo $kambing->Beranak();  ?> |
| 7. | **Latihan 7.7** |  | <?php  class Employee  {      private $first\_name;      private $last\_name;      private $age;      public function \_\_construct($first\_name, $last\_name, $age)      {          $this->first\_name = $first\_name;          $this->last\_name = $last\_name;          $this->age = $age;      }      public function getFirstName()      {          return $this->first\_name;      }      public function getLastName()      {          return $this->last\_name;      }      public function getAge()      {          return $this->age;      }  }  // Membuat object Employee pertama  $objEmployeeOne = new Employee('Bob', 'Smith', 30);  echo $objEmployeeOne->getFirstName(); // prints 'Bob'  echo "<br/>";  echo $objEmployeeOne->getLastName();  // prints 'Smith'  echo "<br/>";  echo $objEmployeeOne->getAge();       // prints '30'  echo "<br/><br/>";  // Membuat object Employee kedua  $objEmployeeTwo = new Employee('John', 'Smith', 34);  echo $objEmployeeTwo->getFirstName(); // prints 'John'  echo "<br/>";  echo $objEmployeeTwo->getLastName();  // prints 'Smith'  echo "<br/>";  echo $objEmployeeTwo->getAge();       // prints '34'  ?> |
|  | **Soal Tes Praktikum 7** |  | <?php  // Class Induk  class Tabungan {      protected $saldo;      public function \_\_construct($saldoAwal) {          $this->saldo = $saldoAwal;      }      public function setor($jumlah) {          $this->saldo += $jumlah;      }      public function tarik($jumlah) {          if ($jumlah <= $this->saldo) {              $this->saldo -= $jumlah;          } else {              echo "Saldo tidak mencukupi!\n";          }      }      public function getSaldo() {          return $this->saldo;      }  }  // Class Anak  class Siswa extends Tabungan {      private $nama;      public function \_\_construct($nama, $saldoAwal) {          parent::\_\_construct($saldoAwal);          $this->nama = $nama;      }      public function getNama() {          return $this->nama;      }  }  // Buat array siswa  $siswaList = [      new Siswa("Siswa 1", 100000),      new Siswa("Siswa 2", 150000),      new Siswa("Siswa 3", 200000),  ];  // Menu interaktif di terminal  while (true) {      echo "\n=== Program Tabungan Sekolah ===\n";      echo "1. Lihat Saldo\n";      echo "2. Setor Tunai\n";      echo "3. Tarik Tunai\n";      echo "4. Keluar\n";      echo "Pilih menu (1-4): ";      $menu = trim(fgets(STDIN));      if ($menu == 4) {          echo "Terima kasih!\n";          break;      }      // Pilih siswa      echo "Pilih siswa:\n";      foreach ($siswaList as $index => $siswa) {          echo ($index + 1) . ". " . $siswa->getNama() . "\n";      }      echo "Masukkan nomor siswa: ";      $pilih = trim(fgets(STDIN)) - 1;      if (!isset($siswaList[$pilih])) {          echo "Siswa tidak ditemukan!\n";          continue;      }      $siswaAktif = $siswaList[$pilih];      switch ($menu) {          case 1:              echo "Saldo " . $siswaAktif->getNama() . " = Rp " . $siswaAktif->getSaldo() . "\n";              break;          case 2:              echo "Masukkan jumlah setor: ";              $jumlah = trim(fgets(STDIN));              $siswaAktif->setor($jumlah);              echo "Setor berhasil! Saldo sekarang = Rp " . $siswaAktif->getSaldo() . "\n";              break;          case 3:              echo "Masukkan jumlah tarik: ";              $jumlah = trim(fgets(STDIN));              $siswaAktif->tarik($jumlah);              echo "Saldo sekarang = Rp " . $siswaAktif->getSaldo() . "\n";              break;          default:              echo "Menu tidak tersedia!\n";      }  }  ?> |

**Analisis dan Penjelasan Kode Program dalam Soal Tes Praktikum :**

Program Latihan Studi Kasus Tabungan Sekolah yang dibuat dengan bahasa pemrograman PHP ini dirancang untuk menerapkan konsep Object Oriented Programming (OOP), khususnya encapsulation, serta dijalankan di terminal (command prompt) dengan input menggunakan fgets(). Program terdiri dari dua kelas utama, yaitu class induk Tabungan dan class anak Siswa. Pada class Tabungan, terdapat atribut saldo yang diberi modifier protected, sehingga hanya dapat diakses dari dalam class itu sendiri maupun class turunannya. Class ini juga memiliki metode setor(), tarik(), dan getSaldo() untuk mengelola saldo tabungan, di mana akses data hanya bisa dilakukan melalui fungsi tersebut, bukan secara langsung, sesuai prinsip encapsulation. Selanjutnya, class Siswa merupakan turunan dari Tabungan yang menambahkan atribut private nama serta menyediakan metode getNama() untuk mengambil data nama siswa. Program kemudian membuat array berisi tiga objek siswa (Siswa 1, Siswa 2, dan Siswa 3) dengan saldo awal berbeda-beda.

Di bagian utama program, dibuat menu interaktif menggunakan perulangan while(true) agar pengguna dapat memilih opsi secara berulang hingga memilih keluar. Menu yang tersedia mencakup melihat saldo, menyetor uang, menarik uang, dan keluar dari program. Saat pengguna memilih menu tertentu, program akan meminta input nomor siswa, lalu menjalankan operasi sesuai pilihan. Pada proses setor dan tarik tunai, saldo akan bertambah atau berkurang sesuai jumlah yang dimasukkan, dengan validasi untuk mencegah penarikan melebihi saldo. Program ini juga memanfaatkan array, constructor, perulangan, serta percabangan, sesuai dengan ketentuan soal. Secara keseluruhan, program ini berhasil mengilustrasikan bagaimana encapsulation menjaga keamanan data saldo tabungan, bagaimana class induk dan class anak berinteraksi melalui pewarisan, serta bagaimana konsep OOP dapat diterapkan dalam sistem sederhana seperti simulasi tabungan sekolah dengan beberapa siswa.