**BAB V**

**FUNCTION DAN METHOD**

## **5. 1 Tujuan**

1. Praktikan dapat membedakan function dan method
2. Praktikan dapat memahami konsep dasar dari function dan method
3. Praktikan dapat memahami fungsi dari function dan method
4. Pratikan dapat menjelaskan function dan method pada source code dalam bahasa Java, C#, Python dan PHP
5. Praktikan dapat mengimplementasikan function dan method ke dalam bahasa Java, C#, Python, dan PHP.
6. Pratikan diharapkan lebih mengenal syntax yang digunakan di bahasa pemograman untuk function dan method
7. Mampu menyelesaikan masalah dengan program sederhana yang didalamnya menggunakan Function dan Method.
8. Mampu mendeklarasikan Function dan Method.

## **5. 2 Dasar Teori**

**5.2. 1 Function**

*Function* adalah sebuah struktur. Pada setiap program pasti memiliki satu*function* utama dan merupakan awal dari program berjalan atau CPU melakukan eksekusi dari sekelompok pernyataan yang diberikan. *Function* akan menjadi kepala dari program dimana semua eksekusi pernyataan berawal.

*(*[*https://www.belajarcpp.com/tutorial/cpp/function/*](https://www.belajarcpp.com/tutorial/cpp/function/)*)*

**5.2. 2 Method**

*Method* adalah sebuah blok program terpisah (diluar program utama) yang kita gunakan untuk menyelesaikan masalah khusus. *Method* bertujuana untuk memecah program kompleks menjadi bagian-bagian kecil sehingga nantinya dapat kita gunakan secara berulang-ulang tanpa harus menulis baris kode yang sama.

*(https://kodedasar.com/method-java/)*

**5.2. 3 Penggunaan Function**

*Function* dipanggil dengan menulis nama dari *function* tersebut, dan diikuti argumen (jika ada). Argumen ditulis dalam tanda kurung, dan jika jumlah argumen lebih dari satu, maka diantaranya dipisahkan oleh karakter utama.

Setelah memproses nilai inputan, hampir semua fungsi akan memberikan nilai hasil pemrosesan tersebut (walaupun ada fungsi yang tidak memberikan nilai). Cara fungsi memberikan nilainya ini sering disebut dengan ‘mengembalikan nilai’ (*return a value).* Nilai yang dikembalikan oleh sebuah fungsi dapat ditampung kedalama variabel, atau langsung ditampilkan ke web browser.

*(https://www.duniailkom.com/tutorial-belajar-php-pengertian-fungsi-function-php-dan-cara-penggunaan-fungsi-php/)*

**5.2. 4 Penggunaan method**

*Method* memiliki nilai balik bila *method* tersebut dipanggil, cara pembuatannya dengan menentukan nilai baliknya. Dalam sebuah *method* fungsi untuk mengembalikan nilainya harus menggunakan kata kunci *return*, dan diikuti dengan nilai yang dikembalikannya.

*(*[*https://bahasa-pemrograman-java.blogspot.com/2014/03/class-di-java.html*](https://bahasa-pemrograman-java.blogspot.com/2014/03/class-di-java.html)*)*

## **5. 3 Hasil Percobaan dan Analisis**

**5.3. 1 Function pada Java**

public class functionjava {

// non-return type dengan parameter

static void cetak(String kelompok){

System.out.println(kelompok);

}

// non-return type tanpa parameter

static void cetak1(){

System.out.println("Hello Praktikan DKP 2020 ^^ ");

}

// return type dengan parameter

static String cetak2(String text){

return text;

}

// return type tanpa parameter

static String cetak3(){

return "Kali ini kita akan belajar mengenai Function";

}

public static void main(String[] args) {

//penggunaan function non-return type dengan megisikan argument untuk parameter kelompok

cetak("Kelompok 52");

//penggunaan function non-return type tanpa parameter

cetak1();

//penggunaan function return type dengan mengisikan nilai pada parameter text

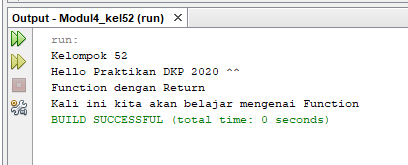
System.out.println(cetak2("Function dengan Return"));

//penggunaan function return type tanpa parameter

System.out.println(cetak3());

}

}



Gambar 5. 1 Hasil dari Function pada Java

Analisis:

Pada source code diatas menggunakan function. Dimana fungsi *Non Return Type* yaitu tidak akan mengembalikan nilai baru dari proses yang dilakukan, fungsi tersebut juga tidak diperlukan *Return*. Kemudian dilanjutkan dengan *Return Type* diikuti dengan Return kemudian Cetak () berfungsi untuk memanggil fungsi cetak.

**5.3. 2 Method pada Java**

public class pendefinisi {

public void greeting(){

System.out.println("Hai, saya method dari class pendefinisi. Salam kenal :)");

}

public void kelompok(String kelompok){

System.out.println(kelompok);

}

public String kenalan (String nama, String hobi){

return "Hai, Nama saya " + nama + " hobi saya " + hobi;

}

}

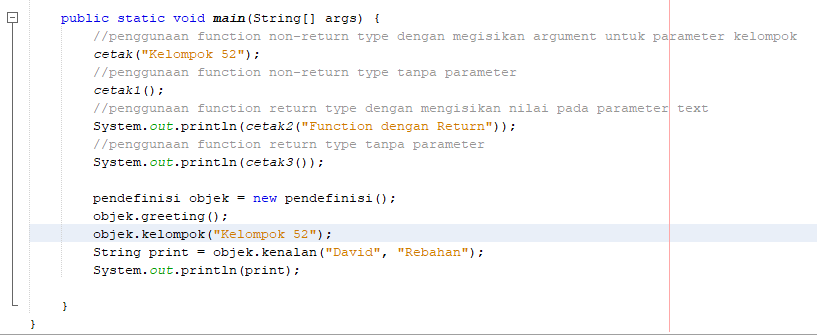
pendefinisi objek = new pendefinisi();

objek.greeting();

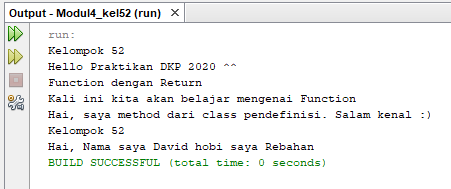
objek.kelompok("Kelompok 52");

String print = objek.kenalan("David", "Rebahan");

System.out.println(print);



Gambar 5. 2 Source Code pada file modul4\_kel3



Gambar 5. 3 Hasil Method pada Java

Analisis:

Pada source code diatas yaitu *Method* pendefinisi yang dibuat pada *new class*. Dimana *method* pendefinisi akan dipanggil dengan cara menambahkan komentar kemudian ditambahkan source code yang ke 2 lalu fungsi pendefinisi objek = new pendefinisi(); ini makusdnya class tadi akan dipanggil ke function awal. kemudian *class* pendefinisi muncul function awal.

**5.3. 3 Function pada C#**

using System;

class Program

{

static void non\_return(String a, String b, String c)

{

Console.WriteLine("Selamat Datang di Praktikum DKP 2020 {0} dan {1} adalah {2}", a, b, c);

}

static int return\_func(int a)

{

if (a > 0 || a < 2) {

return a \* 2;

} else

{

return a \* 0;

}

}

static void Main(string[] args)

{

non\_return("David", "Farizan", "Kelompok52");

Console.WriteLine("Ini adalah praktikum shift {0}", return\_func(1));

Console.ReadKey();

}

}



Gambar 5. 4 hasil function pada C#

Analisis:

Pada source code diatas menggunakan function dengan tipe nilai *non Return type* dengan parameter atau argument. Kemudian diberikan pernyataan dan return\_func() yaitu akan memanggil fungsi. Kemudian diberikan pengkondisian *if* jika a > 0 atau a < 2 dan return a \* 2 jika tidak benar maka return a \* 0**.** Selanjutnya diberikan perintah dan hasil pun tercetak.

**5.3. 4 Method pada C#**

using System;

class Program

{

static void Main(string[] args)

{

percobaan2 objek2 = new percobaan2();

objek2.pembagian(10,5);

int hasil = objek2.pengurangan(10,5);

Console.WriteLine("Hasil pengurangan 10 - 5 = " + hasil);

}

}

// METHOD

class percobaan2

{

public void pembagian(float a, float b)

{

double c = a / b;

Console.WriteLine("Hasil pembagian {0} dengan {1} adalah {2}", a, b, c);

}

public int pengurangan(int a, int b)

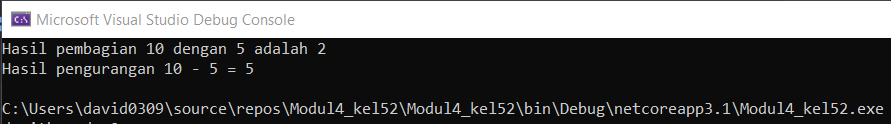
{

int c = a - b;

return c;

}

}



Gambar 5. 5 hasil Method pada C#

Analisis:

Pada program diatas, dibuat terlebih dahulu file *class* yang nantinya akan terdapat *method* didalamnya. Kemudian ada method pembagian dengan parameter *float* a dan *float* b. dan ada juga method pengurangan dengan parameter atau argument *int* a b. method dideklarasikan dengan percobaan2 objek2 = new percobaan2(); yang nantinya akan memanggil class yang sudah ada. objek2.pembagian(10,5); digunakan untuk memanggil *method* pembagian dan objek2.pengurangan(10,5); digunakan untuk memanggil *method* pengurangan. Kemudian diberikan perintah dan hasil tercetak sesuai dengan gambar diatas.

**5.3. 5 Function pada Python**

#Function dengan non return type

def non\_return\_func(praktikan1, praktikan2):

print(f"Selamat Datang di Praktikum DKP 2020 {praktikan1} dan {praktikan2}")

#Function dengan return type

def return\_func(shift):

print(f"Shift kalian adalah ", shift)

if (shift == 1) or (shift == 2) :

return print(f"Fungsi return mengembalikan nilai menjadi ", shift \* 1)

else:

return print("Tidak ada shift tersebut")

#Function dengan Arbitrary Type

def arbitrary\_func(\*penutup):

for nama in penutup:

print("Terimakasih", nama)

#Anonymous Function

anonim\_func = lambda praktikan1, praktikan2, kelompok: print(f"Ini adalah percobaan {praktikan1} dan {praktikan2} kelompok ", kelompok )

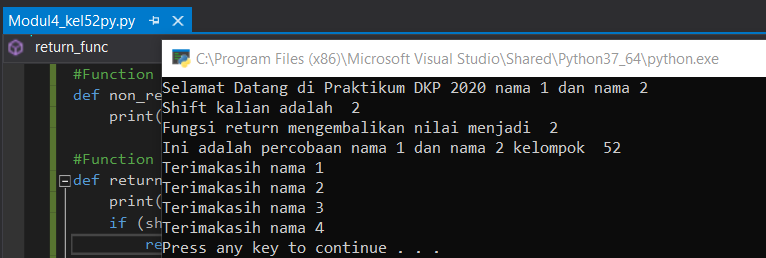
#Pemanggilan Fungsi

non\_return\_func("nama 1", "nama 2")

return\_func(2)

anonim\_func("nama 1", "nama 2", 52)

arbitrary\_func("nama 1", "nama 2", "nama 3", "nama 4")



Gambar 5. 6 hasil Function pada Python

analisis:

pada program diatas menggunakan *function* dengan tipe *non return function, return function, function* dengan *tipe arbitrary type* dan *anonymous function*. *Non return type* dengan parameter praktikan 1 & 2 kemudian *print* pernyataan. Lalu, *function return type* parameter shift lalu pengkondisian *if else*. Selanjutnya *function arbitrary type* dengan adanya penambahan symbol \* ketika pendeklarasian parameter. Digunakan ketika kita tidak mengetahui parameter secara pasti. Kemudian *anonymous function* dapat digunakan tanpa memberi nama fungsinya. Ditandai dengan lamda. Di akhir yaitu pemanggilan fungsi dari setiap tipe nya.

**5.3. 6 Method pada Python**

import percobaan2

p1 = percobaan2.contoh\_method("David", "Farizan")

p1.mulai()

p1.selesai(3)

class contoh\_method:

#init method

def \_\_init\_\_(self, praktikan1, praktikan2):

self.praktikan1 = praktikan1

self.praktikan2 = praktikan2

#self parameter

def mulai(self):

print(f"Selamat Datang di Percobaan 2 {self.praktikan1} dan {self.praktikan2}")

#method dengan parameter

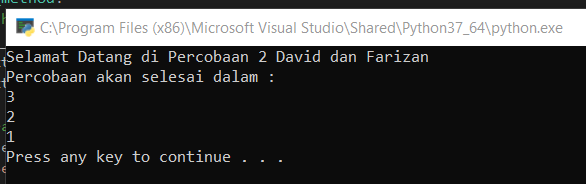
def selesai(self, waktu):

print("Percobaan akan selesai dalam :")

while waktu > 0:

print(waktu)

waktu -= 1



Gambar 5. 7 hasil Method pada Python

analisis:

pada program diatas menunjukan *method*. Setelah itu dibuat *new class* pada *main class* yang gunanya untuk menampung *method*. Lalu pada python terdapat *method init* dengan parameter self, praktikan 1 & 2. *Method* ini akan dieksekusi secara otomatis ketika *class* pada *method* sudah dipanggil. Kemudian *self parameter* yang merupakan *method* pertama dan *method* dengan parameter yang merupakan *method* akhir. Lalu ada perulangan *while* dengan kondisi waktu > 0. Lalu print dan tercetak hasil.

**5.3. 7 Function pada PHP**

<?php

//ini return type

function hitung ($bil1, $bil2) {

return $bil1 + $bil2 \* $bil2;

}

$bil1 = 7;

$bil2 = 8;

$hasil = hitung($bil1, $bil2);

echo "ini dikerjakan oleh function hitung <br>";

echo "angka yang akan dihitung adalah $bil1 dan $bil2 <br>";

echo "hasil hitung menggunakan function hitung yaitu : $hasil";

echo "<hr>";

//ini non return type

function hitung2 ($bil1, $bil2) {

$hasil = hitung($bil1, $bil2);

echo "angka yang akan dihitung adalah $bil1 dan $bil2 <br>";

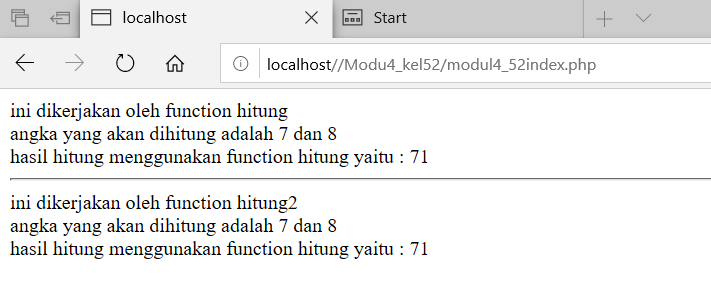
echo "hasil hitung menggunakan function hitung yaitu : $hasil";

}

echo "ini dikerjakan oleh function hitung2 <br>";

hitung2($bil1, $bil2);

?>

  
Gambar 5. 8 hasil *Function* pada PHP

Analisis:

Pada program diatas menggunakan *function return type* dengan parameter bil1 dan bil2, kemudian *return.* Lalu diberikan nilai bil 1 dan 2 dan hasil merupakan hitungan dari bil1 dan 2. Lalu diberikan perintah dan akan ditampilkan ketika di *run*. Kemudian *function non return type* tanpa *return* dan *func* hitung2 dari bil1 dan 2. $hasil = hitung($bil1, $bil2);Yaitu nilai hasil dari type sebelumnya. Lalu hasil keduanya akan didapatkan dari func $hasil”;yang ditunjukkan dari func hitung.

**5.3. 8 Method pada PHP**

<?php

declare(strict\_types = 1);

//class pendefinisi method

class pendefinisi {

public function akar(float $angka) : float {

return sqrt($angka);

}

public function staysafe (string $noun1, string $noun2) {

echo "Jangan lupa cuci tangan dengan $noun1 dan $noun2";

}

}

//code pemanggil method

$angka = 64;

//$objek merupakan instance dari class pendefinisi

$objek = new pendefinisi();

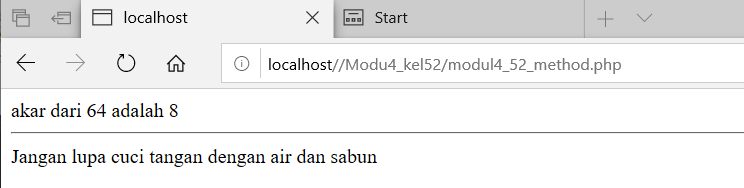
// OOP pada PHP menggunakan -> bukan .

echo "akar dari $angka adalah ".$objek -> akar($angka);

echo "<hr>";

$objek -> staysafe('air', 'sabun');

?>



Gambar 5. 9 hasil *Method* pada PHP

Analisis:

Pada program diatas menggunakan method. Pada php method ditandai dengan panah -> tidak seperti bahasa lain yaitu .method disini yaitu pendefinisi dan pemanggil. file akan dibuat yang berguna untuk menampung method. Kemudian dipanggil dengan $objek -> staysafe('air', 'sabun'); lalu method akan terpanggil dan hasil tercetak

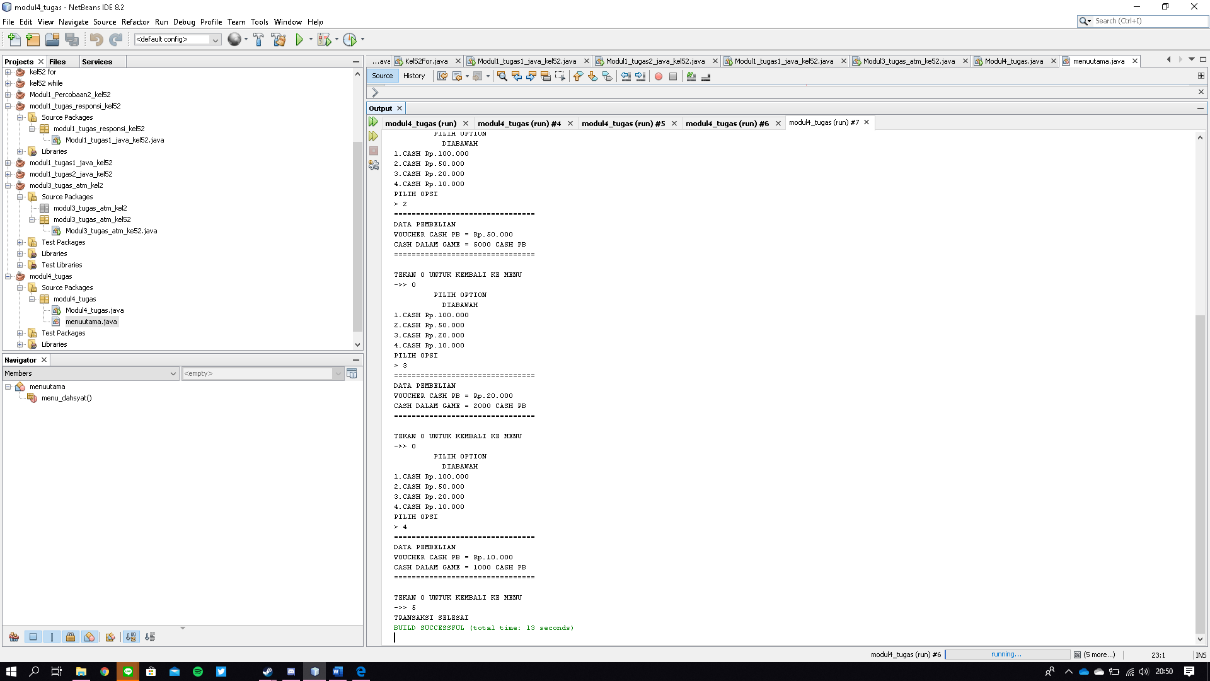
## **5. 4 Tugas**

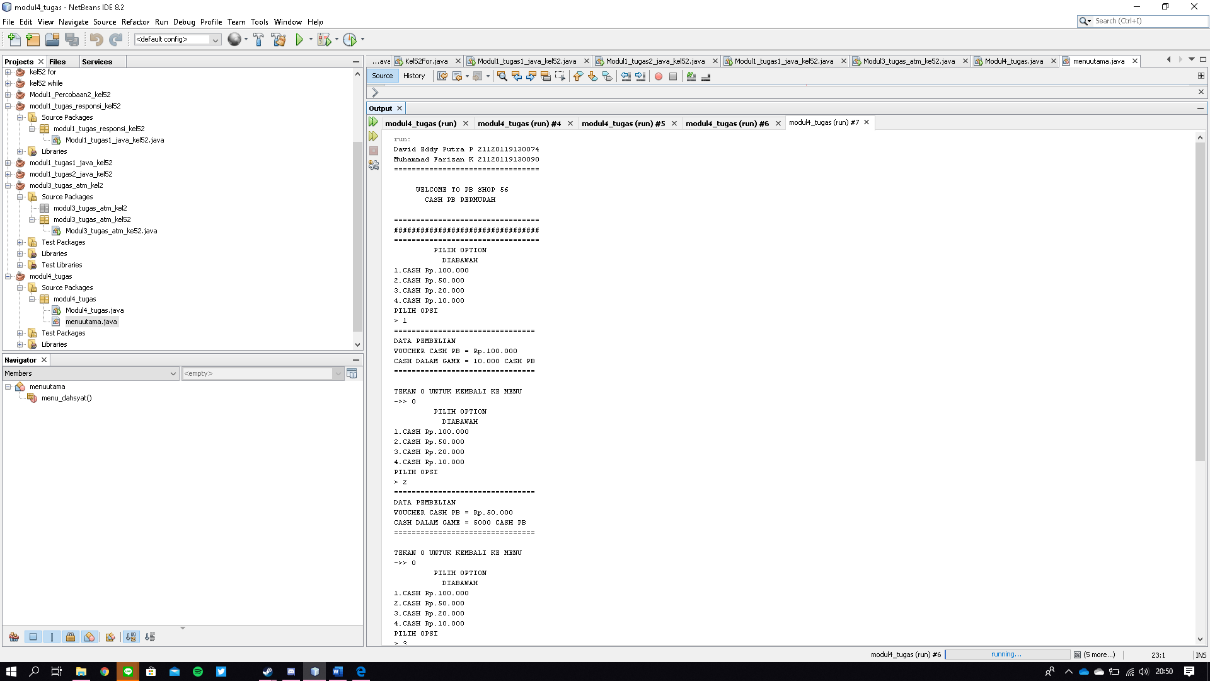
***Source code* pada *main class***

|  |
| --- |
| package modul4\_tugas;  import java.util.Scanner;  /\*\*  \*  \* @author ROG  \*/  public class Modul4\_tugas {  static void orang1(){  System.out.println("David Eddy Putra P 21120119130074");  }  static void orang2(){  System.out.println("Muhammad Farizan K 21120119130090");  }  static void batas(){  System.out.println("=================================");  }  static void menu() {  System.out.println(" WELCOME TO PB SHOP 56 ");  System.out.println(" CASH PB RERMURAH ");  }  static void pagar(){  System.out.println("#################################");  }  /\*\*  \* @param args the command line arguments  \*/  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);{  menuutama objek;  objek = new menuutama();  int proses, option;  orang1();  orang2();  batas();  System.out.println ("");  menu();  System.out.println ("");  batas();  pagar();  batas();    do {  objek.menu\_dahsyat();  System.out.print("> ");  option=sc.nextInt();  switch(option) {  case 1:  System.out.println ("================================");  System.out.println ("DATA PEMBELIAN");  System.out.println ("VOUCHER CASH PB = Rp.100.000");  System.out.println ("CASH DALAM GAME = 10.000 CASH PB");  System.out.println ("================================");  break;    case 2:  System.out.println ("================================");  System.out.println ("DATA PEMBELIAN");  System.out.println ("VOUCHER CASH PB = Rp.50.000");  System.out.println ("CASH DALAM GAME = 5000 CASH PB");  System.out.println ("================================");  break;    case 3:  System.out.println ("================================");  System.out.println ("DATA PEMBELIAN");  System.out.println ("VOUCHER CASH PB = Rp.20.000");  System.out.println ("CASH DALAM GAME = 2000 CASH PB");  System.out.println ("================================");  break;    case 4:  System.out.println ("================================");  System.out.println ("DATA PEMBELIAN");  System.out.println ("VOUCHER CASH PB = Rp.10.000");  System.out.println ("CASH DALAM GAME = 1000 CASH PB");  System.out.println ("================================");  break;  default:  System.out.println("TERIMAKASIH MENGGUNAKAN JASA KAMI");  }  System.out.println ("\nTEKAN 0 UNTUK KEMBALI KE MENU");  System.out.print ("->> ");  proses = sc.nextInt();  ++proses;  }  while ( proses <= 1);  System.out.println("TRANSAKSI SELESAI");  }    }  } |

***Source code* pada *class* menuutama**

|  |
| --- |
| package modul4\_tugas;  /\*\*  \*  \* @author ROG  \*/  public class menuutama {  static void menu\_dahsyat(){  System.out.println(" PILIH OPTION ");  System.out.println(" DIABAWAH ");  System.out.println("1.CASH Rp.100.000 ");  System.out.println("2.CASH Rp.50.000 ");  System.out.println("3.CASH Rp.20.000 ");  System.out.println("4.CASH Rp.10.000 ");  System.out.println("PILIH OPSI");  }  } |

****Gambar 5.10 *output* tugas modul 4

****

Gambar 5.11 *output* tugas modul 4

Analisis:

Jadi ini adalah program untuk pembelian voucher game, diawali pada *source code main class* disana diawali dengan *function static void* orang1,orang2, batas, menu, dan pagar. Lalu kita masukkan *Scanner sc = new Scanner(System.in)* tapi jangan lupa untuk *import java.util.Scanner;* terlebih dahulu agar dapat menggunakan yang diatas tadi.Lalu kita memakai menuutama objek; objek = new menuutama();agar dapat menggunakan *method* pada menuutama *class*.Selanjutnya ada pendeklarasian *integer* pada proses dan *option*, lalu memanggil *function* yang sudah dibuat sebelumnya, lalu menggunakan perulangan *do while* yang diperintahkan untuk mengambil *method* yang sudah dibuat, lalu memasukkan data untuk *variable option*. Selanjutnya ada pengkondisian *switch case* dengan menampilkan 4 pilihan dan *default*. Lalu ada menampilkan penawaran untuk Kembali ke menu dengan menekan angka 0, dengan adanya while yang apabila *variable* proses yang dimasukkan kurang dari 1, maka akan Kembali ke menu utama, namun apabila memasukkan diatas 1, maka akan diselesaikan programnya.

## **5. 5 Kesimpulan**

1. *Function* merupakan *main class* dari suatu fungsi. Apabila tidak ada *function* maka *method* tidak dapat dijalankan
2. Pemanggilan *class* lain dapat dilakukan dengan adanya penambahan *New* pada suatu fungsi yang menitikberatkan pada object. Sehingga *class* akan ditambah.
3. Pemanggilan *class* lain dari *method* menitikberatkan pada object. Ditandai dengan titik (.) seperti object.
4. Pada PHP, *method* tidak ditandai dengan titik (.) melainkan dengan symbol panah (->)
5. Pada *function*, tipe return atau *non void type* diikuti dengan keyword *return* pada fungsinya.
6. *Function* ­*non-return type* menggunakan tipe data *void* dan tidak perlu mengembalikan nilai
7. Pemanggilan *function* bisa langsung mengetikkan nama *function* tersebut (nama*funcion*();)
8. Penulisan *method* diikuti dengan *modifier*