Praktikum 4

Control Flow dan Looping

Tujuan

Memahami logika percabangan dan dapat memakai percabangan dengan tepat.
Memahami logika perulangan dengan tepat dan dapat memberikan kondisi perulangan yang tepat.

Dasar Teori

A. Control Flow (Percabangan)

Percabangan di dalam Java terdapat 2 macam, yaitu dengan memakai if dan switch. Percabangan **`if`** dipakai jika kita menginginkan suatu pernyataan itu dilakukan dengan syarat tertentu yang bernilai benar. Sintaks dari if adalah sebagai berikut:

```
if (ekspresi_boolean) {
       Pernyataan1;
}
```

Pernyataan1 akan dilakukan kalau ekspresi_boolean bernilai true.

Percabangan **`if-else`** dipakai untuk mengeksekusi salah satu dari 2 pernyataan dari syarat tertentu yang pada pada if yang dapat bernilai benar atau salah. Sintaks dari if-else adalah sebagai berikut:

```
if (ekspresi_boolean) {
       Pernyataan1;
} else {
       Pernyataan2;
}
```

Pernyataan1 akan dilakukan kalau ekspresi_boolean bernilai true. Kalau ekspresi_boolean bernilai false, maka Pernyataan2 akan dikerjakan.

Percabangan **`else-if`** dipakai untuk memberikan kondisi tertentu pada bagian **`else`**. Sintaks dari else-if adalah sebagai berikut:

```
if (ekspresi_boolean1) {
       Pernyataan1;
} else if (ekspresi_boolean2) {
       Pernyataan2;
}
```

Ketika ekspresi_boolean bernilai false, maka alur program akan menuju ke bagian else. Selanjutnya Pernyataan2 diatas akan dikerjakan kalau ekspresi_boolean2 bernilai true.

Percabangan **switch** dipakai pada saat kita ingin memberikan kondisi dengan beberapa syarat yang identik yang masing-masing mempunyai pernyataan yang berbeda-beda. Pada Java, nilai yang dilewatkan pada switch harus bertipe int, short, byte atau char. Sintaks dari switch adalah sebagai berikut:

```
switch (ekspresi) {
       case nilai1: Pernyataan1;
       break;
       case nilai2: Pernyataan2;
       break;
       default: Pernyataan3;
}
```

Ketika ekspresi bernilai nilai1, maka alur program akan mengeksekusi Pernyataan1. Selanjutnya break menyebabkan alur program keluar dari daerah switch. Kalau ekspresi bernilai nilai2, maka alur program akan mengeksekusi Pernyataan2. Apabila ekspresi mempunya nilai yang tidak sama dengan nilai1 dan nilai2, maka alur program akan menuju ke bagian **default** dan kemudian mengeksekusi Pernyataan3.

B. Looping (Perulangan)

Perulangan di dalam Java terdapat 3 macam, yaitu for, while dan do-while. Perulangan for dipakai pada saat kita melakukan perulangan dengan jumlah yang sudah diketahui pasti. Sintaks dari for adalah sebagai berikut:

```
for (inisialisasi; kondisi; perubah) {
       Pernyataan;
}
```

Perulangan while dipakai pada saat kita melakukan perulangan dengan jumlah yang belum diketahui pasti. Pernyataan pada while akan dikerjakan setelah pengecekan kondisi pada while bernilai true. Sintaks dari while adalah sebagai berikut:

```
while (kondisi) {
       Pernyataan;
}
```

Perulangan do-while dipakai pada saat kita melakukan perulangan dengan jumlah yang belum diketahui pasti. Pernyataan pada do akan dikerjakan terlebih dahulu, baru setelah itu dilakukan pengecekan kondisi pada while. Sintaks dari do-while adalah sebagai berikut:

```
do {
Pernyataan;
} while (kondisi);
```

Kita dapat memberikan kondisi tertentu pada saat terjadi perulangan. Kondisi yang mungkin terjadi pada perulangan terdapat 2 macam, yaitu **break** dan **continue**. **Break** menyebabkan suatu kondisi untuk keluar dari perulangan. Sedangkan **continue** memyebabkan suatu kondisi untuk melanjutkan ke tahapan selanjutnya pada perulangan.

A. Control Flow

Percobaan 1:

Percobaan berikut ini menunjukkan pemakaian “If-Else” untuk membandingkan nilai numerik.

```
public class IfElseDemo {

    public static void main(String[] args) {
        int hasil=76;
        char grade;

        if(hasil >= 90){
            grade='A';
        }
        else if(hasil >= 80){
            grade='B';
        }
        else if(hasil >= 70){
            grade='C';
        }
        else if(hasil >= 60){
            grade='D';
        }
        else{
            grade='F';
        }
        System.out.println("Grade="+grade);
    }
}
```

Percobaan 2:

Percobaan berikut ini menunjukkan pemakaian “If-Else” untuk membandingkan nilai karakter.

```
public class IfElseName {

    public static void main(String[] args) {
        char initial=(char)-1;
        System.out.println("Enter ur first initial:");

        try{
            initial = (char)System.in.read();
        }
        catch(Exception e){
            System.out.println("Error"+e.toString());
        }

        if(initial == 'a')
            System.out.println("Your name must be Abdullah!");
        else if(initial == 'b')
            System.out.println("Your name must be Bilal!");
        else if(initial == 'c')
            System.out.println("Your name must be Cintania!");
        else
            System.out.println("I cant figure out your name!");
```

```
    }
}
```

Percobaan 3:

Percobaan berikut ini menunjukkan pemakaian “Switch-Case” tanpa “break” untuk membandingkan karakter.

```
public class SwitchName1 {

    public static void main(String[] args) {
        char initial =(char)-1;
        System.out.println("Enter ur first initial:");

        try{
            initial = (char)System.in.read();
        }
        catch(Exception e){
            System.out.println("Error"+e.toString());
        }

        switch(initial){
            case 'a':
                System.out.println("Your name must be Abdullah!");
            case 'b':
                System.out.println("Your name must be Bilal!");
            case 'c':
                System.out.println("Your name must be Cintania!");
            default:
                System.out.println("I cant figure out your name!");
        }
    }
}
```

Percobaan 4:

Percobaan berikut ini menunjukkan pemakaian “Switch-Case” dengan “break” untuk membandingkan karakter.

```
public class SwitchName1 {

    public static void main(String[] args) {
        char initial = (char)-1;
        System.out.println("Enter your rank:");

        try{
            initial =(char)System.in.read();
        }
        catch(Exception e){
            System.out.println("Error"+e.toString());
        }

        switch(initial){
            case '1':
                System.out.println("Your name must be 1st rank!");
                break;
            case '2':
                System.out.println("Your name must be 2nd rank!");
```

```
                break;
            case '3':
                System.out.println("Your name must be 3rd rank!");
                break;
            default:
                System.out.println("I don't know about your rank!");
        }
    }
}
```

Percobaan 5:

Percobaan berikut ini menunjukkan pemakaian "Switch-Case" dengan "break" untuk membandingkan nilai variabel bertipe integer.

```
public class SwitchDemo {
    public static void main(String[] args) {

        int month = 3;
        switch(month){
            case 1:             System.out.println("Sunday");break;
            case 2:             System.out.println("Monday");break;
            case 3:             System.out.println("Tuesday");break;
            case 4:             System.out.println("Wednesday");break;
            case 5:             System.out.println("Thursday");break;
            case 6:             System.out.println("Friday");break;
            case 7:             System.out.println("Saturday");break;
        }
    }
}
```

Percobaan 6:

Percobaan berikut ini menunjukkan pemakaian "If-Else dan "Switch-Case" dengan "break" maupun tanpa "break" untuk pemberian nilai pada jumlah hari tiap bulannya.

```
public class SwitchDemo2 {

    public static void main(String[] args) {
        int month=2;
        int year=2000;
        int numDays=0;

        switch(month){
            case 1:
            case 3:
            case 5:
            case 7:
            case 8:
            case 10:
            case 12:
                numDays=31;
                break;
            case 4:
            case 6:
            case 9:
            case 11:
                numDays=30;
```

```
                break;
            case 2:
                if((year%4==0 &&!(year%100==0))||(year%400==0))
                    numDays=29;
                else
                    numDays=28;
                break;
            default:
                System.out.println("Invalid month");
                break;
        }

        System.out.println("Number of Days ="+ numDays);
    }
}
```

Percobaan 7:

Percobaan berikut ini menunjukkan pemakaian "If-Else dan "Switch-Case" dengan "break" maupun tanpa "break" bersama dengan "argumen" untuk menentukan apakah karakter yang dimasukkan itu "vowel" (huruf vokal) atau bukan.

```
public class VowelCheck {

    public static void main(String[] args){

        if(args.length==0){
            System.out.println("Usage: java VowelCheck <char>");
            System.exit(1);
        }

        char c = args[1].charAt(1);

        switch(c){
            case 'a':
            case 'A':
            case 'i':
            case 'I':
            case 'u':
            case 'U':
            case 'e':
            case 'E':
            case 'o':
            case 'O':
                     System.out.println(c+"is a vowel");
                     break;
            case 'y':
            case 'Y':
                    System.out.println(c+"is sometimes a vowel");
                    break;
            default:
                    System.out.println(c+"isnt a vowel");
        }
    }
}
```

Percobaan 8:

Percobaan berikut ini menunjukkan pemakaian “Switch-Case” dengan beberapa “argumen” untuk mencetak tanggal-bulan-tahun berdasarkan input tanggal.

```
public class SwitchWithArgs {
    public static void main(String[] args) {

        if(args.length!=3){
            System.out.println("argumen <day> <month> <year>");
            System.exit(1);
        }

        int day   =(Integer.valueOf(args[0])).intValue();
        int month =(Integer.valueOf(args[1])).intValue();
        int year  =(Integer.valueOf(args[2])).intValue();

        switch(month){
            case 1: System.out.println(day+"January"+year); break;
            case 2: System.out.println(day+"February"+year); break;
            case 3: System.out.println(day+"March"+year); break;
            case 4: System.out.println(day+"April"+year); break;
            case 5: System.out.println(day+"May"+year); break;
            case 6: System.out.println(day+"June"+year); break;
            case 7: System.out.println(day+"July"+year); break;
            case 8: System.out.println(day+"August"+year); break;
            case 9: System.out.println(day+"September"+year); break;
            case 10: System.out.println(day+"October"+year); break;
            case 11: System.out.println(day+"November"+year); break;
            case 12: System.out.println(day+"December"+year); break;
        }
     }
    }
```

B. Looping

Percobaan 9:

Percobaan berikut ini menunjukkan pemakaian “For” untuk menghitung digit angka.

```
public class ForCounter {
    public static void main (String args[]){

        char input=(char)-1;
        int numToCount;

        System.out.println("Enter a number to count (0-10): ");
        try{
            input=(char)System.in.read();
        }
        catch(Exception e){
            System.out.println("Error:" +e.toString());
        }

        numToCount=Character.digit(input,10);

        if((numToCount>0)&& (numToCount<10)){
            for(int i=1; i<=numToCount; i++)
                System.out.println(i);
        }
        Else{
            System.out.println("That number was not between 0 and 10!");
```

```
        }
    }
}
```

Percobaan 10:

Percobaan berikut ini menunjukkan pemakaian “While” untuk menghitung digit angka.

```
public class WhileCounter {

    public static void main(String[] args) {
        int input =(int)-1;
        int numToCount;

        System.out.println("Enter number to count (1-10): ");
        try{
            input = (int)System.in.read();
        }
        catch (Exception e){
            System.out.println("Error:"+e.toString());
        }

        numToCount = Character.digit(input,10);
        if ((numToCount>0)&&(numToCount<11)){
            int i=1;
            while (i<=numToCount){
                System.out.println(i);
                i++;
            }
        }
        else{
            System.out.println("That number was not beween 0 and 10!");
        }
    }
}
```

Percobaan 11:

Percobaan berikut ini menunjukkan pemakaian “While” dan “DoWhile” untuk menghitung digit angka.

```
public class WhileDemo {
    public static void main (String[]args){
        int count=1;
        while(count<11){
            System.out.println("Count is:"+count);
            count ++;
        }
    }
}

class DoWhileDemo{
    public static void main(String[]args){
        int count = 1;
        do{
            System.out.println("Count is:"+count);
            count++;
        } while (count<=11);
```

```
    }
}
```

Percobaan 12:

Percobaan berikut ini menunjukkan pemakaian “Break” untuk menghentikan looping.

```
public class BreakLoop {
    public static void main (String args[]){
        int i=0;
        do{
            System.out.println("I'm stuck!");
            i++;
            if (i>100) break;
        }
        while(true);
    }
}
```

Percobaan 13:

Percobaan berikut ini menunjukkan pemakaian “Conventional For” dan “Enhanced For” untuk mengakses isi dari array.

```
public class ForDemo {
    public static void main(String[] args){
        int[] numbers= {32, 87, 3, 589, 12, 1076, 2000, 8, 622,127};

        System.out.println(“Data : “

        // Conventional for
        for(int i=0; i < numbers.length; i++){
            System.out.println(numbers[i]+” ”);
        }

        // Enhanced for
        for(int item : numbers){
            System.out.println(“Number of data :”+ item);
        }
    }
}
```

Percobaan 14:

Percobaan berikut ini menunjukkan pemakaian “Continue” pada perulangan.

```
public class ContinueDemo {

    public static void main(String[] args) {
        String searchMe= "peter piper picked a peck of pickled pappers";
        int max = searchMe.length();
        int numPs=0;

        for (int i=0; i<max; i++){
            if(searchMe.charAt(i) != 'p'){
```

```
                continue;
            }
            numPs++;
         }
         System.out.println("Found " + numPs + "p's in the string");
    }
}
```

Percobaan 15:

Percobaan berikut ini menunjukkan pemakaian “Continue” dan “Label” (go to) pada perulangan.

```
public class ContinueDemo2 {

    public static void main(String[] args){
        String searchMe  = "Look for a subString in me";
        String subString = "sub";
        boolean foundIt  = false;
        int max = searchMe.length() - subString.length();

    test:
        for(int i=0; i<=max; i++){
            int n= subString.length();
            int j=1;
            int k=0;

            while(n--!=0){
                if(searchMe.charAt(j++)!= subString.charAt(k++)){
                    continue test;
                }
            }
            foundIt=true;
            break test;
        }
        System.out.println(foundIt ? "Found it" : "Didnt find it" );
    }
}
```

Tugas

1. Buatlah kalkulator sederhana untuk memilih mode hitung *, /, +, -, %, & dan |. Gunakan input parameter dengan contoh :

```
input : 12 * 23
hasil : 276
```

2. Buatlah program untuk menampilkan deret Fibonacci. Contoh tampilan:

```
Masukkan berapa deret Fibonacci? 8
8 deret Fibonacci = 1 1 2 3 5 8 13 21
```