

PERTEMUAN 3

ALGORITMA DAN PEMROGRAMAN DALAM MATEMATIKA

A. Tujuan Pembelajaran

1. Mahasiswa mampu membuat aplikasi sederhana menggunakan perhitungan matematika dengan menerapkan algoritma secara runtunan, pemilihan dan perulangan

B. Uraian Materi

Telah di bahas dalam pertemuan ke-2, bahwasannya **Algoritma** merepresentasikan apa yang diketahui. Sehingga jika tidak ada yang diketahui maka tidak akan ada algoritma.

Dalam bab ini akan menjelaskan tentang implementasi algoritma dalam matematika seperti:

1. Menentukan suatu bilangan genap atau ganjil.
2. Konversi bilangan desimal ke bilangan biner.
3. Menentukan suatu bilangan merupakan bilangan prima.
4. Mencari faktor persekutuan terbesar.
5. Menghitung luas suatu area/bidang yang dibatasi oleh garis

1.1 Menentukan suatu bilangan genap atau ganjil.

Untuk dapat menentukan suatu bilangan termasuk bilangan genap atau bilangan ganjil, kita harus mengetahui konsep dari bilangan genap dan bilangan ganjil.

Bilangan genap adalah bilangan yang habis dibagi 0, atau sisa pembagian dari bilangan tersebut sama dengan 0.

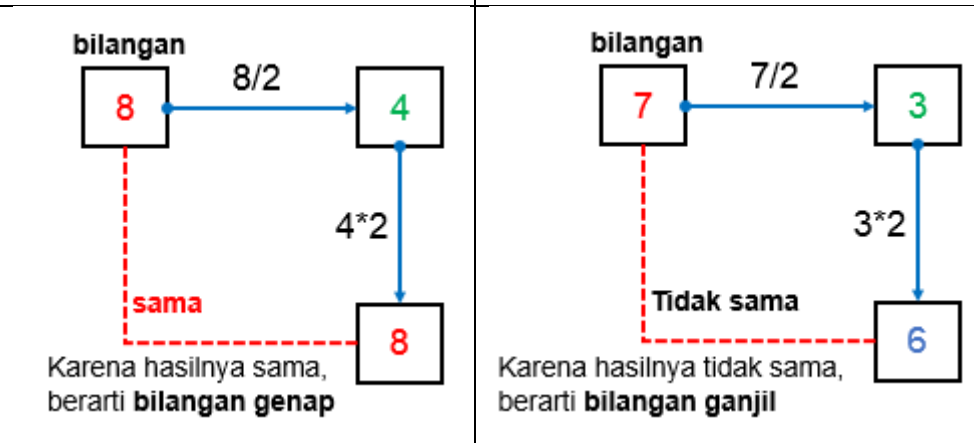
Cara 01	Penjelasan
<pre>cin>> bilangan; if (bilangan % 2 == 0) cout<<"bilangan genap"; else cout<<"bilangan ganjil";</pre>	<p>Contoh : bilangan = 8; 8 % 2 → 4 sisa 0, bilangan genap</p> <p>bilangan = 7; 7 % 2 → 3 sisa 1, bilangan ganjil</p>

Cara 02

```
cin>> bilangan;
if(bilangan == (bilangan/2))
    cout<<"bilangan genap";
else
    cout<<"bilangan ganjil";
```

Penjelasan

apabila bilangan sama dengan bilangan dibagi dua di kali 2 nilai sama dengan bilangan itu sendiri.
Contoh:
Bilangan = 8;
 $(8/2)*2 = 8 \rightarrow$ genap

**1.2 Konversi bilangan desimal ke bilangan biner**

Bila di masukkan suatu bilangan bulat desimal, maka akan menampilkan bilangan dalam bentuk binernya.

Bilangan yang di masukkan

1
5
15
128
255

Yang akan di cetak

0 0 0 0 0 0 0 1
0 0 0 0 0 1 0 1
0 0 0 0 1 1 1 1
1 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1

Cara 1

```
#include<iostream>
using namespace std;
int main(){
    int bil, I, B, A;
    cin>>bil;
    B = 128;
    for(I=1; I<=8; I++)
    {
        if(bil >= B)
        {
            bil = bil - B;
            A = 1;
        }else{
            A = 0;
        }
    }
```


Penjelasan

Nilai maksimal yang dapat dikonversi adalah 255, tersimpan dalam 8 bit bilangan biner yaitu:

1 1 1 1 1 1 1 1

Nilai yang paling kiri pada bilangan biner bernilai 128.

Setiap satu bit akan melakukan satu kali loop, di mana looping maksimum adalah 8 kali $I \leq 8$.

<pre> cout<<A; B = B/2; } return 0; } </pre>	 <p>128/2 = 64 64/2 = 32 Dan seterusnya Digunakan untuk menyatakan nilai-nilai bit.</p>
--	--

Cara 2

Penjelasan

#include<iostream> using namespace std; int main(){ int bil, I, B, A; cin>>bil; B = 128; for(I=1; I<=8; I++) { A = bil / B; cout<<A; bil = bil % B; B = B / 2; } return 0; }	I	N	X	A=bil/B	bil=bil%B	B=B/2
	1	175	128	1	47	64
	2	47	64	0	47	32
	3	45	32	1	15	16
	4	15	16	0	15	8
	5	15	8	1	7	4
	6	7	4	1	3	2
	7	3	2	1	1	1
	8	2	1	1	0	0

bil % B artinya **bil modulus B**, modulus adalah sisa pembagian.
Apabila nilai yang di masukkan 175 akan mencetak:
1 0 1 0 1 1 1 1

1.3 Menentukan suatu bilangan merupakan bilangan prima.

Bilangan prima adalah bilangan bulat yang habis dibagi jika hanya di bagi dengan bilangan itu sendiri, kecuali angka 1.

Contoh bilangan prima :

2 3 5 7 11 17 19 23 29 dan seterusnya

terdapat beberapa konsep pemikiran untuk menentukan bilangan prima:

konsep pertama:

Misalnya **bil** merupakan suatu bilangan yang di masukkan, maka akan diperiksa, apakah Bil akan habis di bagi dengan salah satu nilai yang ada di bawahnya atau nilai yang lebih kecil dari nilai **bil**

Apabila habis → maka Bil bukan bilangan prima

Apabila tidak habis → maka Bil merupakan bilangan prima.

Contoh :

Misal → **bil** = 9

Bilangan yang ada di bawah nilai 9 adalah : 2, 3, 4, 5, 6, 7, 8

Jika kita bagi nilai 9 dengan nilai atau bilangan yang ada di bawahnya, ternyata 9 habis di bagi dengan 3. Sehingga 9 bukan bilangan prima.

Misal → **bil** = 11

Bilangan yang ada di bawah nilai 11 adalah : 2, 3, 4, 5, 6, 7, 8, 9, 10, 11

Jika kita bagi 11 dengan nilai atau bilangan yang ada di bawahnya ternyata 11 tidak habis di dengan bilangan yang ada di bawahnya.

Sehingga 11 merupakan bilangan prima.

Konsep pertama untuk menentukan bilangan prima, jika di imlementasikan ke dalam program menggunakan bahasa pemrograman C++, dapat di lihat pada listing program berikut:

Listing 14.1: bil prima1.cpp

```
#include<iostream>
using namespace std;
main()
{
    int bil, I, tanda;
    tanda = 0;
    cin>>bil;
    I = 2;
    while(I<=bil-1)
    {
        if(bil % I == 0)
            tanda = 1;
        I++;
    }
    if(tanda == 1)
        cout<<"bukan bilangan prima";
    else
        cout<<"bilangan prima";
}
```



Untuk program di atas:

jika bilangan yang di masukkan 9 maka akan mencetak

bukan bilangan prima

jika bilangan yang di masukkan 11, maka akan mencetak

bilangan priman

Konsep Kedua:

Kita ambil contoh bilangan prima berikut: 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, ...

Bila kita perhatikan bilangan prima diatas selain angka 2, maka semua bilangan **prima** merupakan bilangan **ganjil**.

Algoritma yang dapat kita buat untuk memeriksa apakah **bil** merupakan bilangan **prima** atau **bukan**, dapat dibuat sebagai berikut:

1. Jika **bil** = 2, maka cetak "**bilangan PRIMA**" dan proses selesai
2. Jika **bil** = bukan 2, maka periksa :
 - a. Jika **bil** bilangan genap (bilangan genap → habis dibagi 2), maka cetak "**BUKAN PRIMA**" dan proses selesai.
 - b. Jika **bil** bilangan ganjil, maka periksa apakah bilangan tersebut prima atau bukan.

Listing 14.2: bil prima2.cpp

```
#include<iostream>
#include<math.h>
using namespace std;
main()
{
    int bil, batas, X, tanda;
    cin>>bil;
    if(bil==2)
        cout<<"bilangan PRIMA";
    else{
        if(bil%2 == 0)
            cout<<"BUKAN PRIMA";
        else{
            X = 3;
            batas = bil-1;
            tanda = 0;
            while(tanda == 0 && X <= batas)
            {
                if(bil % X == 0)
                    tanda = 1;
                X = X + 2;
            }if(tanda == 0)
                cout<<"bilangan PRIMA";
            else
                cout<<"BUKAN PRIMA";
        }
    }
}
```

1.4 Mencari faktor persekutuan terbesar

Faktor Persebutuan Terbesar (FPB) adalah pembagi yang mempunyai nilai terbesar. Perhatikan contoh berikut:

Misalnya diketahui dua buah bilangan bulat n_1 dan n_2 , masing masing bernilai:

$n_1 = 30$ dan $n_2 = 105$

untuk mencari faktor persekutuan terbesar dari kedua bilangan tersebut di lakukan cara berikut:

30 habis dibagi dengan: **1** 2 **3** **5** 6 12 **15** 30

90 habis dibagi dengan: **1** **3** **5** 7 **15** 21 35 105

Jika kita lihat hasil di atas bahwa nilai 30 dan 105 mempunyai pembagi habis yang sama: **1, 3, 5, 15**.

Dan nilai pembagi terbesar adalah **15**, jadi faktor persekutuan terbesar dari 30 dan 105 adalah: **15**.

Cara diatas adalah cara yang biasa digunakan dalam mencari faktor persekutuan terbesar.

Menggunakan Algoritma Euclidean

Algoritma Euclides adalah implementasi atau penerapan algoritma berulang-ulang atau berkali-kali sampai menghasilkan sisa yang sama dengan nol (sisa = 0).

Misal diketahui dua buah bilangan bulat $n_1 = 105$ dan $n_2 = 30$, dengan menggunakan **euclidean algoritm** dilakukan dengan cara berikut:

$105 = (3) (30) + 15 \rightarrow 3$ didapat dari pembagian ($105/30 = 3$) \rightarrow dan 15 adalah sisanya

$15 = (1) (15) + 0 \rightarrow 15$ di dapat dari sisa pembagian sebelumnya, 1 didapat dari pembagian ($15/15 = 1$) \rightarrow sisanya adalah 0.

Jadi faktor pesekutuan terbesar dari 105 dan 30 adalah 15 (merupakan nilai pembagi terakhir).

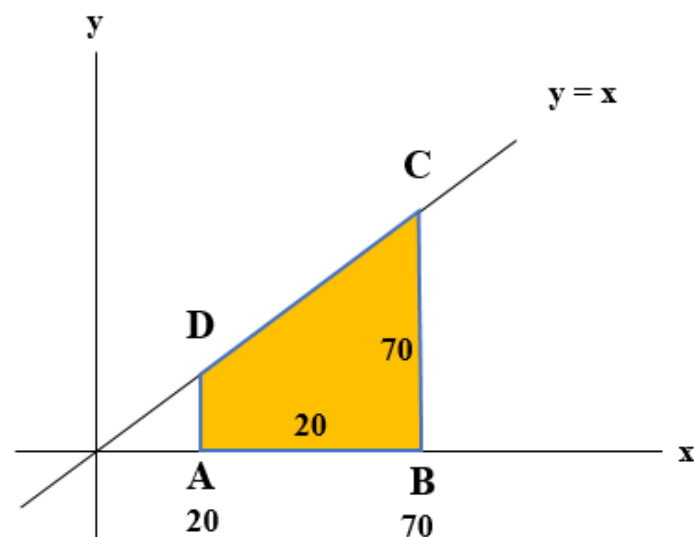
Implementasi Euclidean Algoritm ke dalam bahasa pemrograman C++, dapat di lihat dengan progam berikut:

Listing 14.3: fpb.cpp

```
#include<iostream>
using namespace std;
main()
{
    int n1, n2, x, sisa;
    cin>>n1>>n2;
    if(n1<n2)
    {
        x=n1; n1=n2; n2=x;
    }
    while(n2 != 0)
    {
        sisa = n1%n2;
        n1=n2;
        n2 = sisa;
    }
    cout<<"FPT : "<<n1;
}
```

1.5 Menghitung luas suatu area/bidang yang dibatasi oleh garis

Sebagai contoh terdapat sebuah trapesium ABCD dengan gambar sebagai berikut:



Trapezium di atas di bentuk oleh empat buah garis:

$$y = 0$$

$$y = x$$

$$x = 20$$

$$x = 70$$

secara matematika luas trapesium dapat dihitung dengan

$$= (70 + 20) \cdot 50 / 2$$

$$= 2250$$

Jika menggunakan integral:

$$\begin{aligned} \text{Luas trapesium} &= \int_{20}^{70} x \, dx \\ &= \left[\frac{1}{2} x^2 \right]_{20}^{70} \\ &= 0.5 \cdot 4900 - 0.5 \cdot 400 \\ &= 0.5 \cdot 4500 \\ &= 2250 \end{aligned}$$

Listing 14.4: luastrapesium.cpp

```
#include<iostream>
using namespace std;
main()
{
    float awal = 20, akhir=70, dx, LuasTotal;
    float Luasdx, X, Y;

    dx = (akhir-awal)/50;
    LuasTotal= 0;
    while(awal<akhir){
        X = awal+(0.5*dx);
        Y=X;
        Luasdx = Y * dx;
        LuasTotal = LuasTotal + Luasdx;
        awal = awal + dx;
    }
    cout<<"Luas Trapesium : "<<LuasTotal;
}
```



Jika program diatas dijalankan akan mencetak:

2250

C. Soal Latihan / Tugas

1. Susun program untuk mengkonversi bilangan biner ke bilangan desimal ?
2. Susun algoritma dan buat program untuk memasukkan dua buah bilangan bulat positif yang berbeda, kemudian cetak nilai persekutuan kelipatan terkecil (KPK) kedua buah bilangan tersebut.
3. Susun algoritma dan buat program untuk mencetak bilangan segitiga pascal dengan memanfaatkan atau menggunakan array bedimensi satu.

D. References

- Lestari, F. D. (2017). Analisa Algoritma Faktor Persekutuan Terbesar (FPB) Menggunakan Bahasa Pemrograman C++. *Jurnal Evolusi* , 63-68.
- Sjukani, M. (2014). *Algoritma dan Struktur Data 1 dengan C, C++ dan Java* (Edisi 9 ed.). Jakarta: Mltra Wacana Media.
- Yuniati, S. (2012). MENENTUKAN KELIPATAN PERSEKUTUAN TERKECIL (KPK) DAN FAKTOR PERSEKUTUAN TERBESAR (FPB) DENGAN MENGGUNAKAN METODE "PEBI". *Beta*, 149-165.