LAPORAN PRAKTIKUM
METODE NUMERIK

Judul: Galat/Error

DISUSUN OLEH

ILHAM NUR ROMDONI M0520038

PROGRAM INFORMATIKA

FAKULTAS MIPA

UNIVERSITAS SEBELAS MARET

2021

SCREENSHOT

A. Screenshot Praktikum

1. Program 1: Menghitung nilai $\sqrt{2}$

```
% Ilham Nur Romdoni, M0520038

% Menghitung nilai akar 2 secara eksak
A = sqrt (2)

% Menghitung nilai akar 2 dengan rumus 1
x=1;
e=1;
while e > 0.00001,
    y=x;
    x= (y+2/y)/2
    e=abs(x-y);
end
fprintf('%5.7f\n',x);
```

a. Menghitung nilai galat dari nilai eksak dengan nilai dari metode.

```
% Ilham Nur Romdoni, M0520038

% Menghitung nilai akar 2 secara eksak
A = sqrt(2)

% Menghitung nilai akar 2 dengan rumus 1
x=1;
e=1;
while e > 0.0000001,
    y=x;
    x=(y+2/y)/2
    e=abs(x-y)
end
fprintf('x = %5.15f\n',x);
E= abs(x-A);
fprintf('e = %5.15f\n', E);
```

b. Menghitung nilai galat dari nilai eksak dengan nilai dari metode jika batas iterasi menjadi 0,5.

```
% Ilham Nur Romdoni, M0520038

% Menghitung nilai akar 2 secara eksak
A = sqrt(2)

% Menghitung nilai akar 2 dengan rumus 1
x=1;
e=1;
while e > 0.5,
    y=x;
    x=(y+2/y)/2
```

```
    e=abs(x-y)
end
fprintf('x = %5.15f\n',x);
E= abs(x-A);
fprintf('e = %5.15f\n', E);
```

2. **Program 2: Menghitung e^x**

```
% Ilham Nur Romdoni, M0520038

x = input('Input nilai x : ');
% Menghitung nila e^x dengan nilai eksak
A = exp(x)

% Menghitung nilai e^x dengan deret Taylor
n = input('Input nilai n : ');
B = 1;
for i = 1:n,
    B = B + (x^2/factorial(i))
end
B
```

```
% Ilham Nur Romdoni, M0520038

function f = factorial(m)
f = 1;
for i = m:-1:1,
    f = f*i;
end
```

a. Menghitung nilai *error* dari nilai e^1 dengan Deret Taylor, misalkan n = 10 atau 5.

```
% Ilham Nur Romdoni, M0520038

x = input('Input nilai x : ');
% Menghitung nilai e^x dengan nilai eksak
A = exp(x)

% Menghitung nilai e^x dengan deret Taylor
n = input('Input nilai n : ');
B = 1;
for i = 1:n,
    B = B + (x^2/factorial(i));
end
B

e=abs(A-B);
fprintf('%5.15f', e)
```

B. Screenshot Source Code

1. Soal No 1

Tentukan nilai galat atau *error* pada perhitungan nilai f(x) = $\sqrt{2}$ dengan rumus $x_n = \frac{1}{2}(x_{n-1} + \frac{2}{x_{n+1}})$ dengan $n \geq 2$ dan $x_n = 1$ dengan $n = 1$.

```
% Ilham Nur Romdoni, M0520038

% Menghitung nilai f(x)
A = sqrt(2)

% Menghitung nilai akar dengan rumus xn
n = input('n = ');
x = 1;
for i = 1:n,
    y = x;
    x = 1/2*(y+2/y);
end
fprintf('x = %5.15f\n',x);
E= abs(x-A);
fprintf('e = %5.15f\n', E);
```

2. Soal No 2

Tentukan nilai galat atau *error* pada perhitungan nilai

$$f(x) = \sqrt{1} + \sqrt{2} + \sqrt{3} + \cdots + \sqrt{99} + \sqrt{100}$$

dengan ketiga metode di bawah ini:

a. Perhitungan secara eksak.

b. Masing-masing akar dikalikan 100 dan dibulatkan.

c. Tanpa *looping* (Menggunakan fungsi SUM dalam MATLAB).

```
% Ilham Nur Romdoni, M0520038

% a. Perhitungan secara eksak
a = 0;
for i = 1:100,
    a = a + sqrt(i);
end;
fprintf('a = %5.15f\n',a);

% b. Masing-masing akar dikalikan 100 dan dibulatkan
b = 0;
for i =1:100,
    b = b + (100*sqrt(i));
    b = round(b);
end;
b = b/100;
E = abs(b-a);
fprintf('b = %5.2f\n',b);
fprintf('e = %5.15f\n', E);
```

```
% c. Tanpa looping
i = 1:100;
c = sum(sqrt(i));
c;
E = abs(c-a);
fprintf('c = %5.15f\n',c);
fprintf('e = %5.15f\n',E);
```

3. Soal No 3

Tentukan nilai galat atau *error* pada perhitungan nilai $f(x) = cos\,(x)$ dengan menggunakan deret Taylor yang dapat dirumuskan di bawah ini:

$$cos(x) = \lim_{N\to\infty} \sum_{n=0}^{N} (-1)^n . \frac{x^{2n}}{2n!}$$

```
% Ilham Nur Romdoni, M0520038

% Menghitung nilai eksak
x = input ('Masukkan nilai x = ');
A = cos(x);
fprintf('cos(x) = %5.15f\n', A)

N = input('Input nilai N : ');

% Menghitung nilai dengan deret taylor
B = 1;
for n = 1:N,
    B = B + ((-1)^n)*(x^(2*n))/factorial(2*n);
end
B

e = abs(A-B);
fprintf('e = %5.15f\n', e)
```

ANALISIS PRAKTIKUM

A. Analisis Source Code

1. Program 1: Menghitung nilai $\sqrt{2}$

Untuk menghitung nilai eksak dari $\sqrt{2}$ pada MATLAB, menggunakan fungsi sqrt. Jadi nilai variabel A akan diisi oleh hasil dari sqrt(2). Selanjutnya untuk melakukan penghitungan nilai dengan rumus 1, dimulai dengan mendefinisikan nilai awal yaitu x=1 dan e =1. Setelah itu melakukan pengulangan while di mana syarat e > 0.00001. Ketika e masih > 0.00001 maka pengulangan while masih berjalan. Variabel y diinisialisasikan dengan nilai x sedangkan nilai x diisi dengan nilai baru dari rumus (y+2/y)/2. Nilai e dihitung dengan abs(x-y). abs adalah fungsi untuk nilai mutlak. Fungsi fprintf digunakan untuk menampilkan nilai dari variabel. %5.7f berarti angka 5 adalah jumlah angka di depan koma desimal. Sedangkan 7 adalah jumlah angka di belakang koma dan huruf f merepresentasikan bahwa bilangan ini bertipe data float.

a. Menghitung nilai galat dari nilai eksak dengan nilai dari metode.
   Mendefinisikan variabel E sebagai nilai galat dari eksak dengan rumus 1. E = abs(x-A) yaitu nilai mutlak dari pengurangan nilai eksak dan nilai rumus pada iterasi terakhir.
b. Menghitung nilai galat dari nilai eksak dengan nilai dari metode jika batas iterasi menjadi 0,5.
   Nilai e pada pengulangan while diganti dengan 0,5.

2. Program 2: Menghitung e^x

Nilai e^x dihitung menggunakan fungsi exp(x) di mana nilai x diambil dari *input*-an *user*. *Input*-an *user* dapat disimpan dengan menggunakan fungsi input(). Penghitungan pendekatan dilakukan dengan deret Taylor. n didefinisikan sebagai batas pengulangan yang akan dilakukan for. Variabel B memiliki nilai awal yaitu 1. Pengulangan menggunakan for di mana didefinisikan pengulangan i = 1 sampai n kali. Variabel B akan diisi dari hasil rumus B + (x^2/factorial(i)) yang mana nilai terus bertambah dan nilai B akan berganti terus sesuai nilai i. Fungsi factorial bukan bawaan dari MATLAB. Fungsi factorial dibuat sendiri dengan mendefinisikan f sebagai factorial(m). Setelah menentukan nilai awal, melakukan pengulangan for dengan i =

m, setiap perulangan berkurang 1. Yaitu dengan menuliskan for I = m:-1:1. Lalu mendefinisikan variabel f dengan f*i.

a. Menghitung nilai *error* dari nilai e^1 dengan Deret Taylor, misalkan n = 10 atau 5.

Mendefinisikan variabel e sebagai nilai galat dari eksak dengan deret Taylor. e = abs(A-B) yaitu nilai mutlak dari pengurangan nilai eksak dan nilai deret Taylor pada iterasi terakhir. Nilai x yang di-*input*-kan adalah 1 dan nilai n yang di-*input*-kan adalah 10 atau 5.

3. Soal No 1

Pendefinisian nilai awalnya, yaitu x = 1. Memberikan *input* ke nilai n. Pendekatan dihitung menggunakan perulangan i yang dimulai dari 1 hingga berhenti di n (sesuai angka *input*-an). Menginisialisasikan variabel y = x. Nilai x diisi dengan nilai yang baru dari rumus ½*(y+2/y). Nilai e yang mendefinisikan *error*, diisi dengan nilai absolut (mutlak) dari x dikurangi y. Nantinya akan menampilkan hasil dari *error* pada perulangan terakhir.

4. Soal No 2

Pendefinisian a = 0. Perulangan i di mana akan mengalami perulangan hingga 100 kali. Inisialisasi a bahwa nantinya variabel a ditambah dengan akar dari perulangan saat itu (i). Menampilkan hasil dari a dengan bilangan desimal 15 digit di belakang koma.

Pendefinisian b = 0. Perulangan i di mana akan mengalami perulangan hingga 100 kali. Inisialisasi b bahwa nantinya variabel b ditambah dengan perkalian antara bilangan 100 dan akar i. Kemudian variabel b dibulatkan menggunakan fungsi round. Perhitungan *error* dengan absolut dari variabel b dikurangi variabel a. Menampilkan hasil b yang berupa desimal dengan 2 digit di belakang koma dan hasil *error* dari b.

Pendefinisian bahwa i nantinya adalah bilangan dari 1 hingga 100. Inisialisasi c sama dengan penjumlahan (sum) dari akar i di mana i tersebut adalah bilangan dari 1 hingga 100. Perhitungan *error* dengan absolut dari variabel c dikurangi variabel a. Menampilkan hasil c dan hasil *error* dari c.

5. **Soal No 3**

Memberikan *input* pada x, dengan fungsi *input*. Pendefinisian variabel A = cos(x) di mana x-nya di dapat dari *input*-an *user*. Kemudian hasil tersebut ditampilkan ke *Command Window* menggunakan fungsi fprintf. Kemudian memberikan *input* pada N. pendefinisian bahwa B = 1 di mana angka awal untuk memulai deret taylor tersebut. Melakukan perulangan n di mana dari 1 hingga ke N (diambil dari *input*-an N dari *user*). Selama perulangan 1 ke N akan menjalankan fungsi B = B + ((-1)^n)*(x^(2*n)). Kemudian nilai B ditampilkan. Menghitung nilai *error* dengan absolut dari variabel A dikurangi dengan variabel B.

B. Analisis Jalannya Program

1. Program 1: Menghitung nilai $\sqrt{2}$

```
>> Hitung_akar1

A =

    1.4142


x =

    1.5000


x =

    1.4167


x =

    1.4142


x =

    1.4142

1.4142136
```

Pada program asli praktikum, menampilkan hasil dari akar 2 dan nilai x dengan batasan error yaitu 0.00001.

a. Menghitung nilai galat dari nilai eksak dengan nilai dari metode.

```
>> Hitung_akar

A =

    1.4142


x =

    1.5000


e =

    0.5000


x =

    1.4167


e =

    0.0833


x =

    1.4142


e =

    0.0025


x =

    1.4142


e =

   2.1239e-06


x =

    1.4142


e =

   1.5949e-12

x = 1.414213562373095
e = 0.000000000000000
```

Program yang dibagikan pada *google classroom*, menggunakan batasan *error* 0.0000001 dan menampilkan hasil x dengan 15 angka desimal. Nilai e = 0.

b. Menghitung nilai galat dari nilai eksak dengan nilai dari metode jika batas iterasi menjadi 0,5.

```
>> Hitung_akar_iterasi_diubah

A =

    1.4142


x =

    1.5000


e =

    0.5000

x = 1.500000000000000
e = 0.085786437626905
```

Error semakin besar karena batas iterasi 0.5 sehingga proses iterasi berlangsung sedikit.

2. Program 2: Menghitung e^x

```
>> Hitung_e1
Input nilai x : 1

A =

    2.7183

Input nilai n : 5

B =

     2


B =

    2.5000

B =

    2.6667

B =

    2.7083

B =

    2.7167

B =

    2.7167
```

Pada program asli praktikum, menampilkan hasil dari nilai exp(x) dan nilai dari deret Taylor sampai iterasi ke-n.

a. Menghitung nilai *error* dari nilai e^1 dengan Deret Taylor, misalkan n = 10 atau 5.

```
>> Hitung_e
Input nilai x : 1

A =

    2.7183

Input nilai n : 10

B =

    2.7183

0.000000027312661>>
>> Hitung_e
Input nilai x : 1

A =

    2.7183

Input nilai n : 5

B =

    2.7167

0.001615161792379>>
```

Dengan n=10 memberikan error 0.000000027312661 sedangkan dengan n=5 memberikan nilai error 0.001615161792379. Semakin banyak iterasi, semakin kecil *error*-nya.

3. Soal No 1

```
>> Praktikum4No1

A =

    1.4142

n = 2
x = 1.416666666666667
e = 0.002453104293571
>> Praktikum4No1

A =

    1.4142

n = 5
x = 1.414213562373095
e = 0.000000000000000
>> Praktikum4No1

A =

    1.4142

n = 10
x = 1.414213562373095
e = 0.000000000000000
```

Program menampilkan nilai *error* dari soal no 1. Ketika n=2, *error* bernilai 0.002453104293571. Sedangkan saat n=5 dan n=10, *error* bernilai 0. Hal ini membuktikan bahwa semakin banyak perulangan, maka nilai *error*-nya semakin kecil bahkan mendekati tanpa *error*. Nilai e = 0 menunjukkan batas maksimal iterasi.

4. Soal No 2

```
>> Praktikum4No2
a = 671.462947103147712
b = 671.48
e = 0.017052896852306
c = 671.462947103147712
e = 0.000000000000000
```

Hasil dari proses perhitungan ditunjukkan bahwa hasil *error* pada 2b yaitu 0.017052896852306 dengan rumus mutlak dari nilai eksak pada poin a dikurangi dengan nilai pendekatan dengan *rounded* atau pembulatan. Sedangkan pada 2c *error* sama dengan nol karena hasil dari perhitungan tanpa perulangan sama dengan nilai eksak sehingga jika dikurangi akan menghasilkan nol.

5. Soal No 3

```
>> Praktikum4No3
Masukkan nilai x = 1
cos(x) = 0.540302305868140
Input nilai N : 4

B =

    0.5403

e = 0.000000273496940
>> Praktikum4No3
Masukkan nilai x = 1
cos(x) = 0.540302305868140
Input nilai N : 100

B =

    0.5403

e = 0.000000000000000
```

Hasil dari percobaan tersebut bahwa pada iterasi 4 memiliki *error* 0.000000273496940 dan iterasi 100 memiliki *error* yang sangat mendekati 0. Sehingga semakin banyak iterasi maka *error* dari suatu perhitungan akan semakin kecil. Nilai e = 0 menunjukkan batas nilai maksimal dari N atau iterasi penyelesaian.