Modul Praktikum

Fisika Komputasi I

Dengan Matlab GUI

IGA Widagda
Fisika FMIPA UNUD
2014

Kata Pengantar

Sebelumnya kami memanjatkan puji syukur kepada Tuhan Yang Maha Esa, karena berkat rahmat-Nya maka kami bisa menerbitkan *Modul Praktikum Fisika Komputasi I dengan Matlab Gui*. Kami berharap semoga buku ini dapat membantu mahasiswa Fisika FMIPA UNUD dalam menguasai materi kuliah Fisika Komputasi I.

Kami juga menghaturkan terima kasih kepada pihak-pihak yang telah membantu dalam penyusunan modul ini :

-Bapak Ir. S. Poniman, M.Si., Ketua Jurusan Fisika FMIPA UNUD

-Teman – teman dosen dan karyawan Jurusan Fisika, FMIPA, UNUD

-Istriku, Kusuma dewi dan kedua anakku, Widya dan Lestari

Kami menyadari bahwa modul praktikum ini masih banyak kekurangannya, untuk itu kami sangat mengharapkan kritik dan saran dari semua pihak untuk kesempurnaan modul ini.

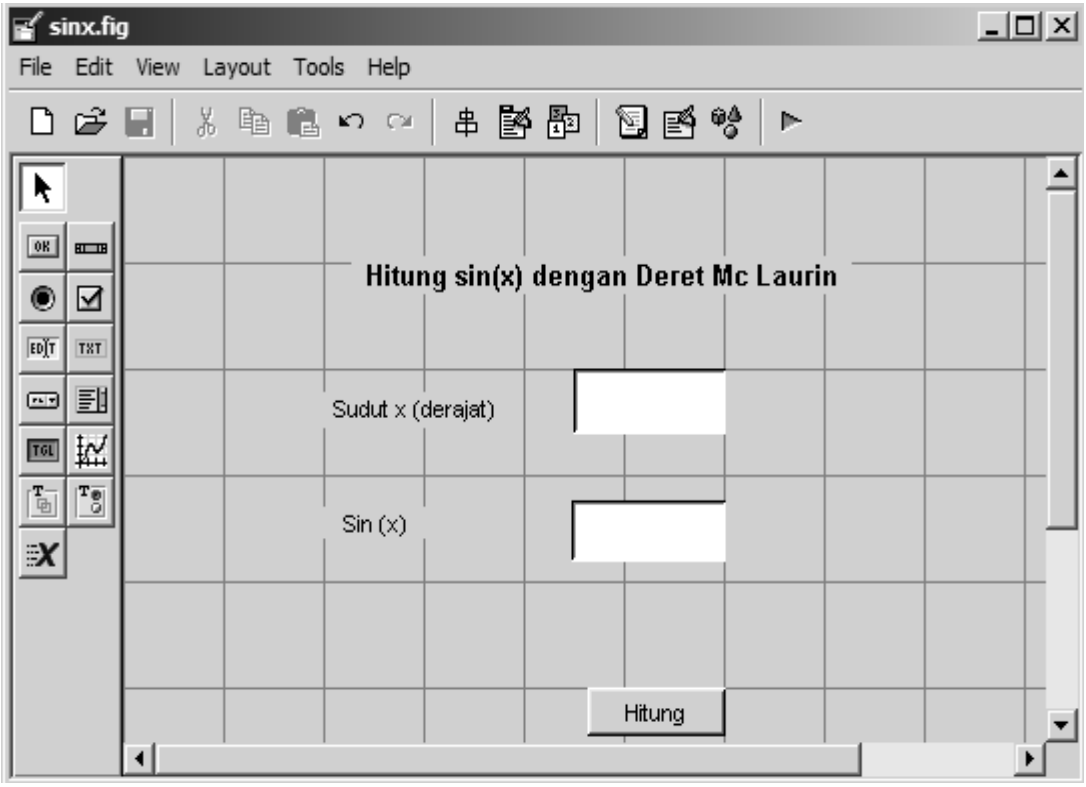
PENYUSUN

DAFTAR ISI

1 Deret Taylor

1.1 Program Komputer Deret Taylor dan Mac Laurin

1.1.1.Rancangan Desain



Gambar 1.1 Desain GUI : Deret Mac Laurin

1.1.2 Daftar Komponen

No	Komponen	Properti	
1	Static text1	String	Sudut x (derajat)
2	Static text2	String	Sin(x)
3	Edit Text1	String	
		Tag	x
4	Edit Text2	String	
		Tag	sinx
5	Pushbutton1	String	Hitung
		Tag	hitung

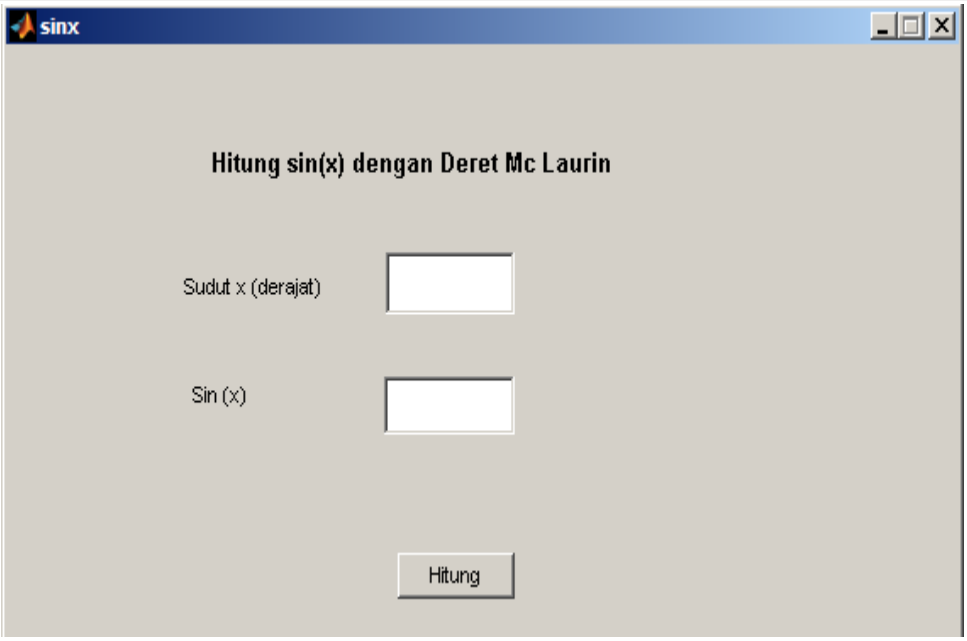
1.1.3. Kode Program (*source code*)

1.1.3.1 Function *hitung_Callback*

```
function hitung_Callback
x = str2num(get(handles.x,'String'));
x=x/57.3; %ubah jadi radian
eps=1e-5;
del=10;
m=0;
while del > eps
        m=m+1;
        s=0;
        j=0;
        for i=1:1:2*m-1
           if mod(i,2) ==1
                    j=j+1;
                    if mod(j,2) == 0
                             tanda = -1;
                    else
                             tanda=1;
                    end
                    pem=1;
                    pen=1;
                    for k=1:1:i
                             pem=pem*x;
                             pen=pen*k;
                      end
                    s=s+tanda*pem/pen; %deret Sinus
           end
           if m>1
                del= abs(sinusx-s);
           end
           sinusx=s;
        end
        set(handles.sinx,'String',num2str(s));
end
```

1.1.4 Hasil Eksekusi (*RUN*) program

Hasil eksekusi program Newton-Rapshon dapat dilihat dalam Gambar 1.2.

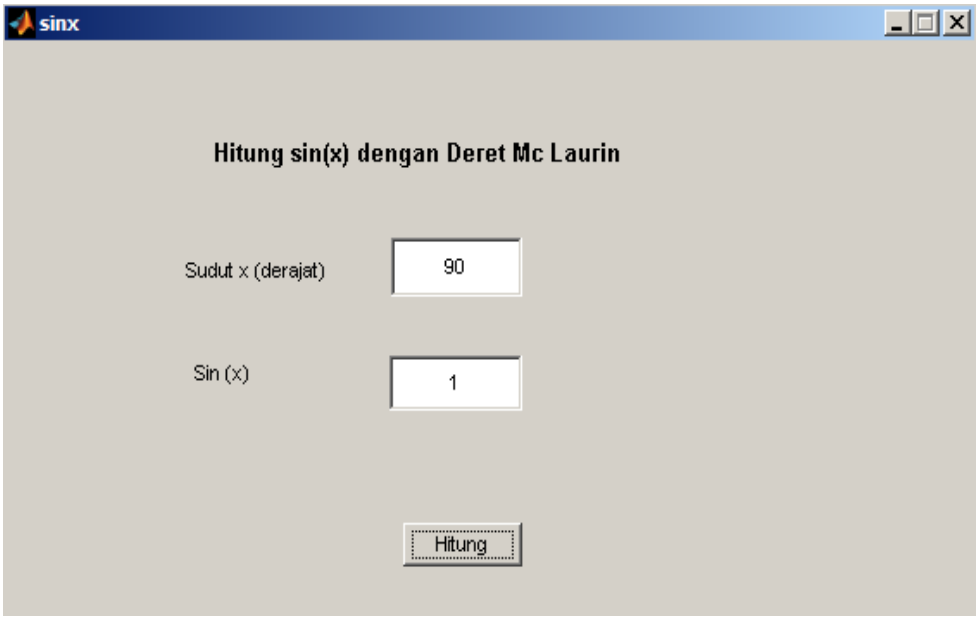


Gambar 1.2 Hasil Eksekusi program Mc Laurin

Cara kerja program :

- Masukkan nilai *Sudut x (derajat) : 90*
- Klik tombol *Hitung*

Maka akan didapatkan hasil seperti diperlihatkan dalam Gambar 1.3.



Gambar 1.3 Hasil Akhir eksekusi program Mc Laurin

Penjelasan hasil program :

-program menampilkan hasil yaitu : *sin (x) = 1*

1.2 Latihan

1. Buatlah program komputer dari deret Mac Laurin untuk fungsi *exp(x)* :

$$\exp(x) = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots$$

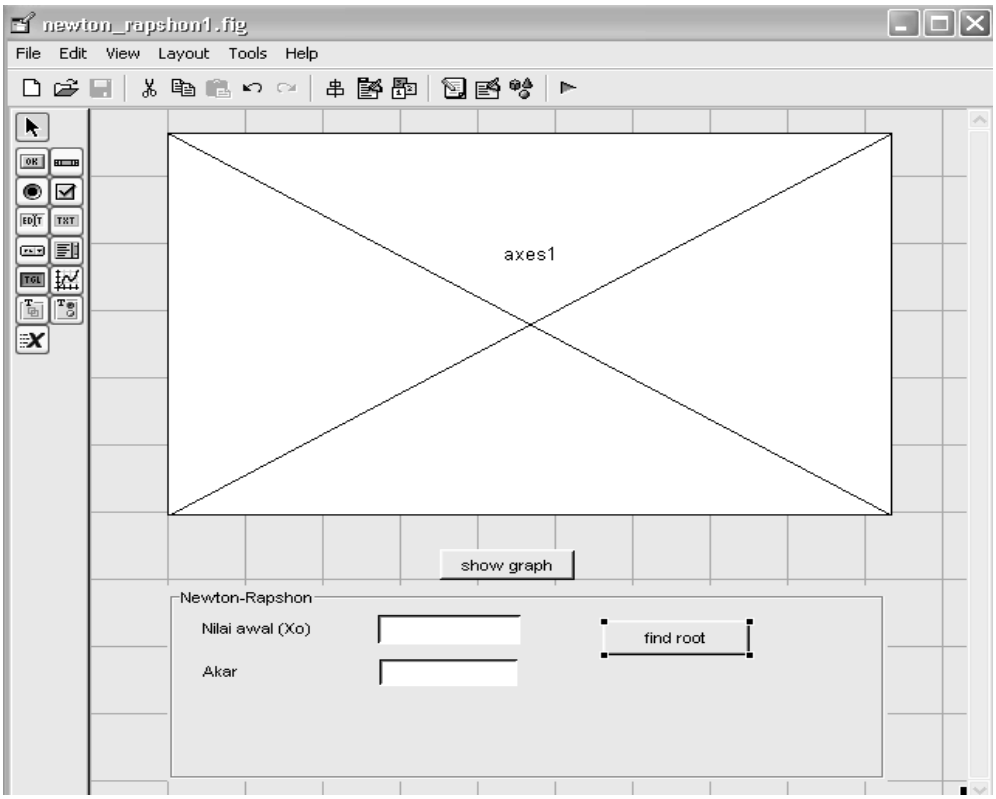
2. Buatlah program komputer dari deret Mac Laurin untuk fungsi *cos(x)* :

$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \ldots$$

2 AKAR

2.1 Program Komputer Newton-Rapshon

2.1.1.Rancangan Desain



Gambar 2.1 Desain GUI : Newton - Rapshon

2.1.2 Daftar Komponen

No	Komponen	Properti	
1	Axes1	Tag	axes1
2	Static text1	String	Nilai awal(Xo)
3	Static text2	String	Akar
4	Edit Text1	String	
		Tag	nilai_awal_edit
5	Edit Text2	String	
		Tag	akar_edit
6	Pushbutton1	String	show graph
		Tag	show_pushbutton
7	Pushbutton2	String	find root
		Tag	find_root_pushbutton

2.1.3. Kode Program (*source code*)

2.1.3.1 Function *show_pushbutton_Callback*

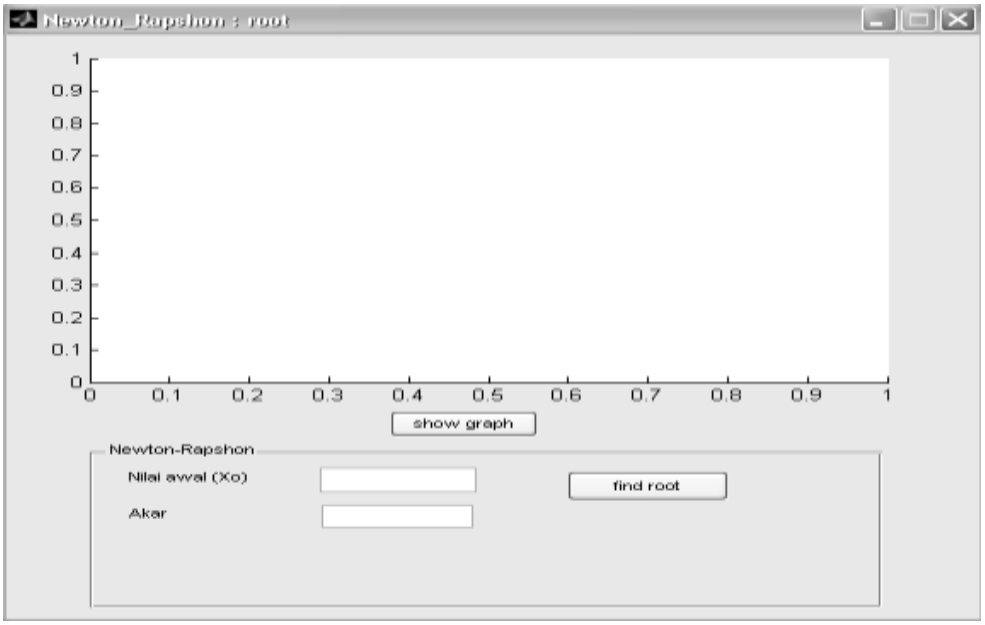
```
function show_pushbutton_Callback
axes(handles.axes1);
x = 0:.1:1000;
a2=0.001;
a1 =1;
y = -a2*x.^2+a1*x;
plot(x,y);
title('grafik y=-0.001.x^2+x');
xlabel('x');
ylabel('y');
```

2.1.3.2 Function : *find_root_pushbutton_Callback*

```
function find_root_pushbutton_Callback(hObject, eventdata, handles)
%*********************************************************
%Hitung akar dari fungsi f(x), dengan nilai awal xo
%*********************************************************
eps=1e-5;
y(1) = str2num(get(handles.nilai_awal_edit,'String'));
n=1;
selisih = 10;
while selisih >= eps
   n=n+1;
   x=y(n-1);
   %fungsi berikut (fx dan dx) dpt berubah sesuai kasus
   %*****************************
     %fx=x^3-11*x^2+39*x-45;       %Fungsi yg dicari akarnya
     %dx=3*x^2-22*x+39;            % Turunan pertama dari fungsi
      fx=-0.001*x^2+x;             %kasus gerak peluru
      dx=-0.002*x;
      %fx=-x+4;
      %dx=-1;
   %*************************
      y(n)=x-fx/dx;                %pers. Metode Newton-Rapshon orde-1
      selisih=abs(y(n)-x);         %selisih 2 nilai iterasi yg berurutan
end
set(handles.akar_edit,'String',num2str(y(n)));
```

2.1.4 Hasil Eksekusi (*RUN*) program

Hasil eksekusi program Newton-Rapshon dapat dilihat dalam gambar 2.2.

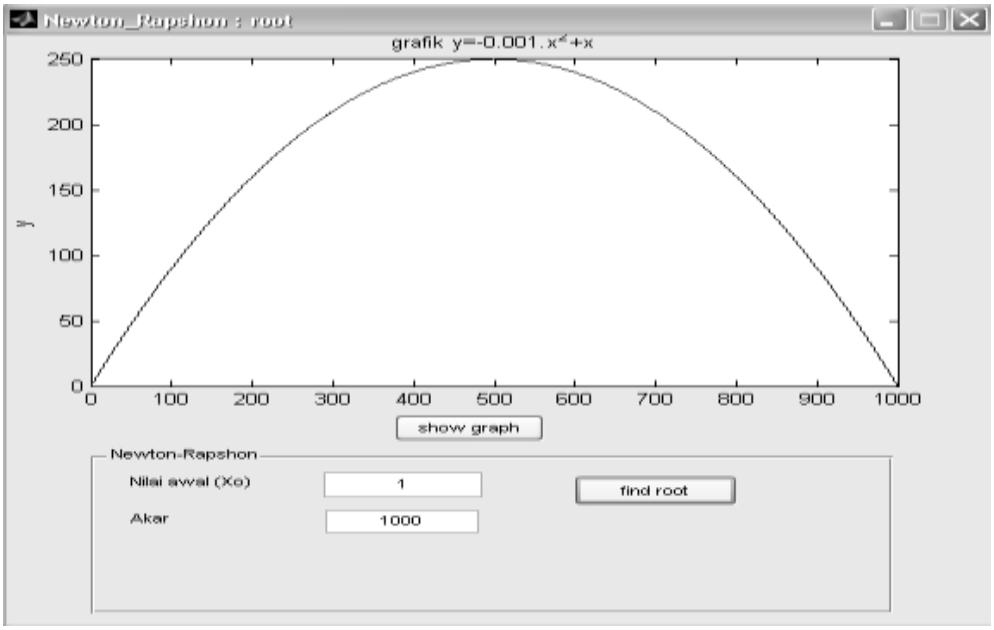


Gambar 2.2 Hasil Eksekusi program Newton-Rapshon

Cara kerja program :

- Klik tombol *show graph*
- Masukkan *Nilai awal*, misal : 1
- Klik tombol *find root*

Maka akan didapatkan hasil seperti diperlihatkan dalam gambar 2.3.



Gambar 2.3 Hasil Akhir eksekusi program Newton-Rapshon

Penjelasan hasil program :

-program menampilkan kurva dari $y = -0.001x^2 + x$

-nilai awal (x0) : 1

Memberi nilai awal 1 pada proses iterasi Newton-Rapshon

-Hasilnya adalah :

Akar = 1000, saat x = 1000 maka y(1000) = 0

2.2 Latihan

1. Carilah akar dari fungsi :

$$y = -x + 4$$

Petunjuk :

-Modifikasi pada Function *show_pushbutton_Callback* :

```
x=0:0.1:10;
y=-x+4;
title('grafik y=-x+4');
```

-Modifikasi pada function find_root_pushbutton_Callback :

```
fx=-x+4;   %bentuk fungsi
dx=-1;     %turunan fungsi
```

2. Carilah akar dari fungsi :

$$y = x^3 - 11x^2 + 39x - 45$$

Petunjuk :

-Modifikasi pada Function *show_pushbutton_Callback* :

```
x=-20:0.1:20;
y =x.^3-11*x.^2+39*x-45;
title('grafik y=x^3-11x^2+39x-45');
```

-Modifikasi pada function find_root_pushbutton_Callback :

```
fx=x^3-11*x^2+39*x-45;      %Fungsi yg dicari akarnya
dx=3*x^2-22*x+39;           % Turunan pertama dari fungsi
```

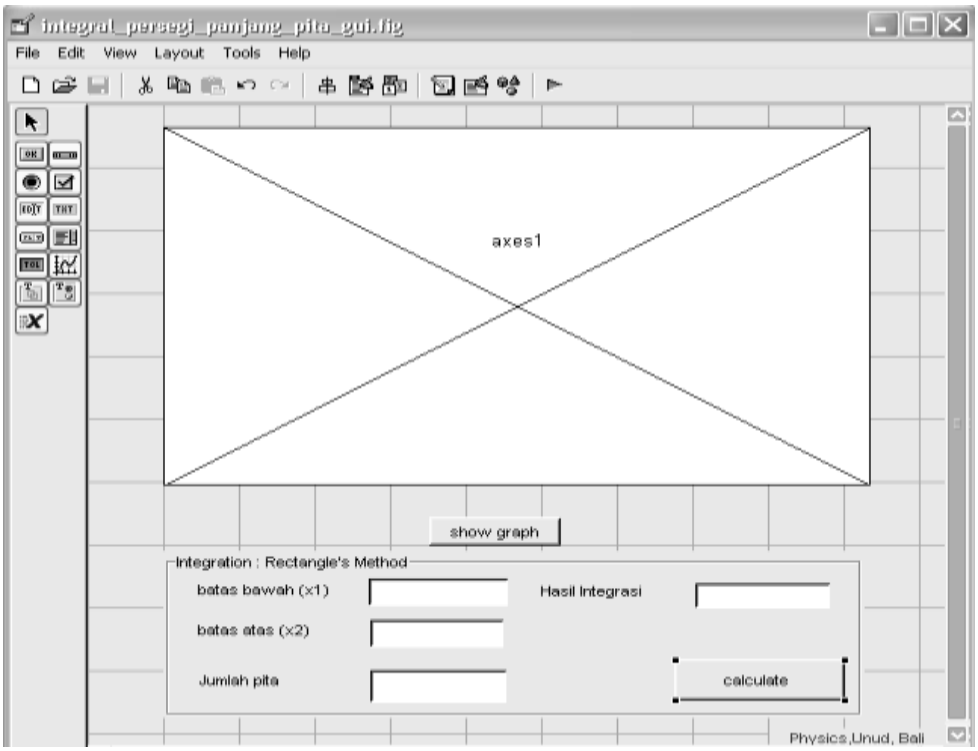
- beri nilai awal (x_0) : 1, maka hasilnya akar =3

  beri nilai awal (x_0) : 6, maka hasilnya akar =5

3 INTEGRAL DAN DIFERENSIAL

3.1 Program Komputer Integral : Metode Persegi Panjang

3.1.1.Rancangan Desain



Gambar 3.1 Desain GUI : Integral Persegi Panjang

3.1.2 Daftar Komponen

No	Komponen	Properti	
1	Axes1	Tag	axes1
2	Static text1	String	batas bawah (x1)
3	Static text2	String	batas atas (x2)
4	Static text3	String	Jumlah pita
5	Static text4	String	Hasil Integrasi
6	Edit Text1	String	
		Tag	batas_bawah_edit
7	Edit Text2	String	
		Tag	batas_atas_edit
8	Edit Text3	String	
		Tag	jml_pita_edit
9	Edit Text4	String	
		Tag	hasil_persegi_panjang_edit
10	Pushbutton1	String	show graph
		Tag	show_pushbutton
11	Pushbutton2	String	calculate
		Tag	calculate_pushbutton
12	Panel1	Title	Integration: Rectangle's Method

3.1.3. Kode Program (*source code*)

3.1.3.1 Function *show_graph_pushbutton_Callback*

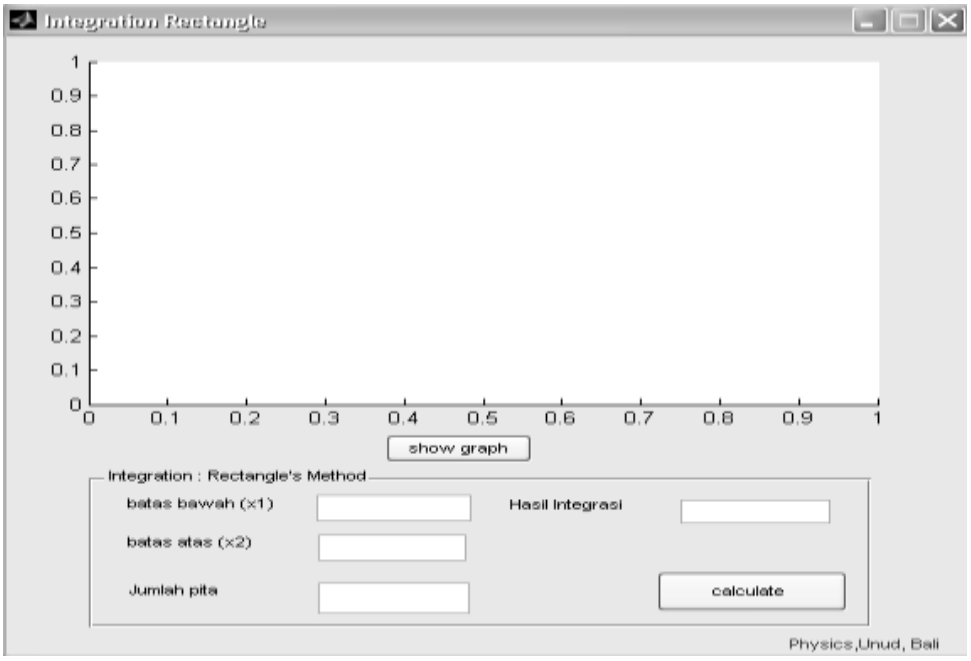
```
function show_graph_pushbutton_Callback(hObject, eventdata, handles)
axes(handles.axes1);
x=0:100;
y=x.^2;
plot(x,y);
title('y=x^2');
xlabel('x');
ylabel('y');
```

3.1.3.2 Function : *calculate_pushbutton_Callback*

```
function calculate_pushbutton_Callback(hObject, eventdata, handles)
x1 = str2num(get(handles.batas_bawah_edit,'String'));
x2 = str2num(get(handles.batas_atas_edit,'String'));
pita = str2num(get(handles.jml_pita_edit,'String'));
persegi=0;
x=x1;
delx=(x2-x1)/pita; %pita= jumlah pita
while x<x2      %delx=lebar masing-masing pita;
   %*********************************
   fx=x^2;  % fungsi yang dihitung
   %*********************************
   persegi=persegi+fx*delx;
   x=x+delx;
end
set(handles.hasil_persegi_panjang_edit,'String',num2str(persegi));
x=0:100;
y=x.^2;
plot(x,y);
title('y=x^2');
xlabel('x');
ylabel('y');
for a=x1:0.01:x2
   Ya=a.^2;
   line([a a],[min(y) Ya],'color','r');
end
```

3.1.4 Hasil Eksekusi (*RUN*) program

Hasil eksekusi program Integral Persegi Panjang dapat dilihat pada gambar 3.2.

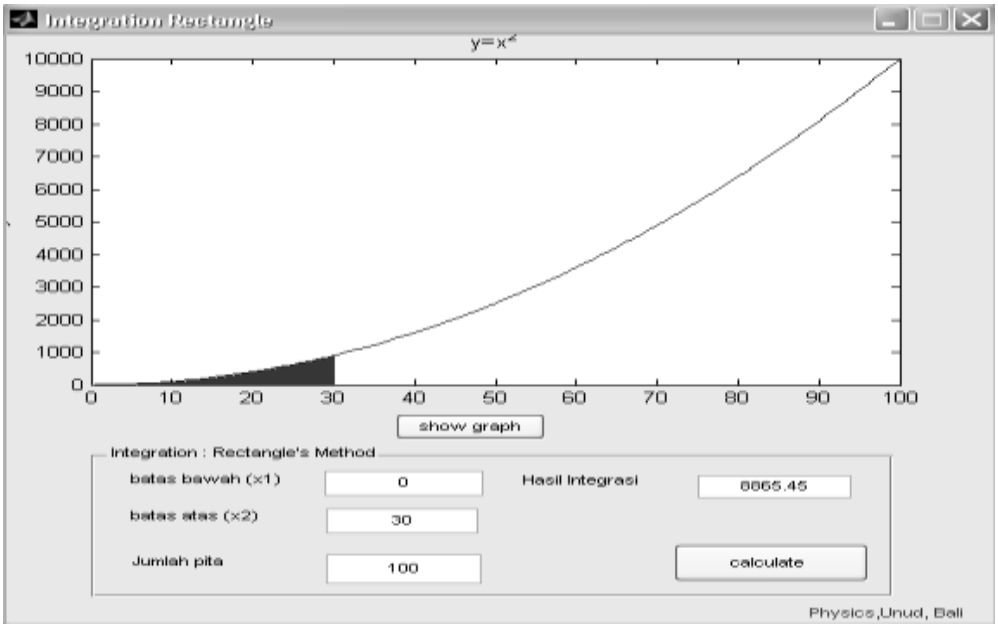


Gambar 3.2 Hasil Eksekusi program Integral Persegi Panjang

Cara kerja program :

- Klik tombol *show graph*
- Masukkan *batas bawah (x1)*, misal : 0
- Masukkan *batas atas (x2)*, misal : 30
- Masukkan *Jumlah pita*, misal : 100

Maka akan didapatkan hasil seperti diperlihatkan dalam gambar 3.3.



Gambar 3.3 Hasil Akhir eksekusi program Integral P.Panjang

Penjelasan hasil program :

-program menampilkan kurva dari $y = x^2$

-batas bawah(x1) : 0

Memberikan batas bawah integral yaitu 0

-batas atas (x2) : 30

Memberikan batas atas integral yaitu 30

-jumlah pita : 100

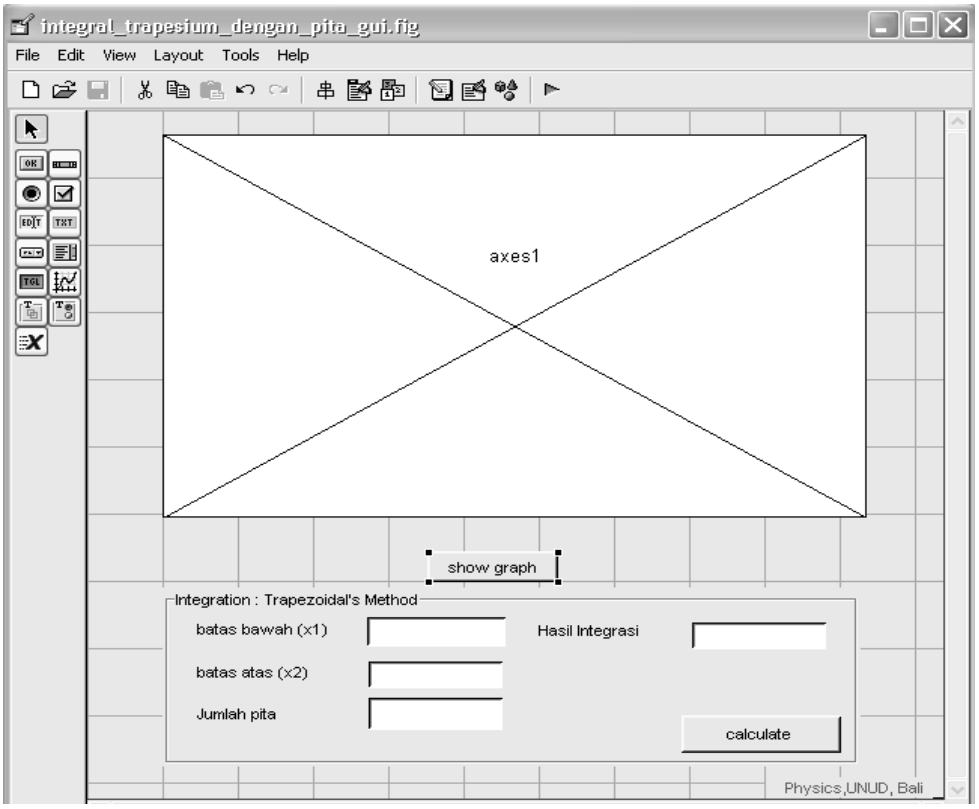
Antara batas bawah(x1) dan batas atas (x2) dibagi menjadi 100 bagian yang masing-masing berbentuk persegi panjang

-Hasil integrasi : 8865.45

Hasil perhitungan integral dengan metode Persegi panjang adalah 8865.45 yang ditunjukkan dengan luas daerah yang diarsir pada kurva gambar 3.3.

3.2 Program Komputer Integral : Metode Trapesium

3.2.1.Rancangan Desain



Gambar 3.4 Desain GUI : Integral Trapesium

3.2.2 Daftar Komponen

No	Komponen	Properti	
1	Axes1	Tag	axes1
2	Static text1	String	batas bawah (x1)
3	Static text2	String	batas atas (x2)
4	Static text3	String	Jumlah pita
5	Static text3	String	Jumlah pita
6	Edit Text1	String	
		Tag	batas_bawah_edit
7	Edit Text2	String	
		Tag	batas_atas_edit
8	Edit Text3	String	
		Tag	jml_pita_edit
9	Edit Text4	String	
		Tag	hasil_trapesium_edit
10	Pushbutton1	String	show graph
		Tag	show_pushbutton
11	Pushbutton2	String	calculate
		Tag	calculate_pushbutton
12	Panel1	Title	Integration : Trapezoidal's Method

3.2.3. Kode Program (*source code*)

3.2.3.1 Function *show_graph_pushbutton_Callback*

```
function show_graph_pushbutton_Callback(hObject, eventdata, handles)
axes(handles.axes1);
x=0:100;
y=x.^2;
plot(x,y);
title('y=x^2');
xlabel('x');
ylabel('y');
```

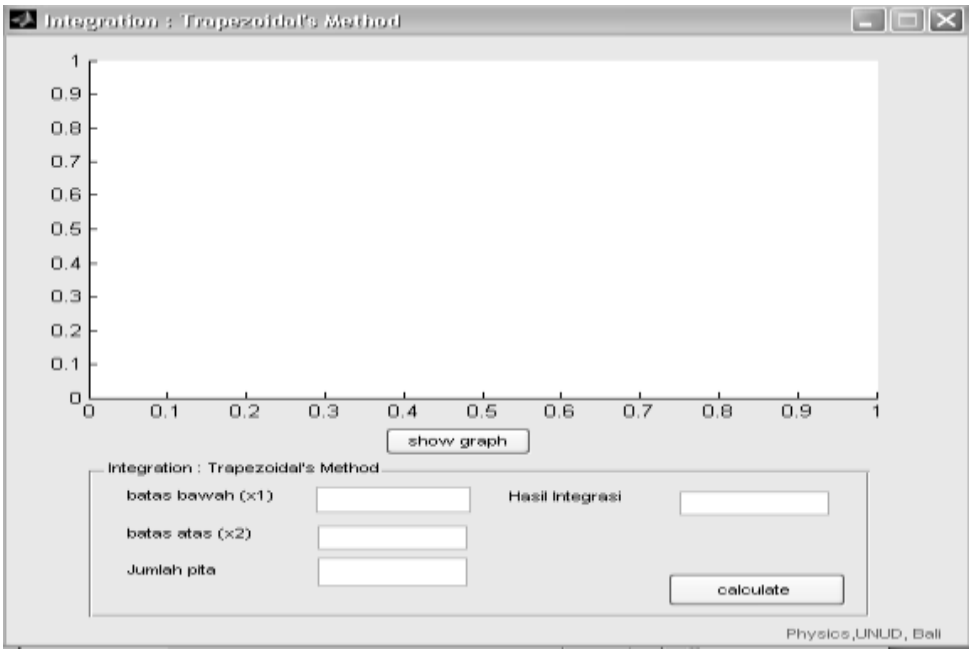
3.1.3.2 Function : *calculate_pushbutton_Callback*

```
function calculate_pushbutton_Callback(hObject, eventdata, handles)
x1 = str2num(get(handles.batas_bawah_edit,'String'));
x2 = str2num(get(handles.batas_atas_edit,'String'));
pita = str2num(get(handles.jml_pita_edit,'String'));
trap=0;
x=x1;
delx=(x2-x1)/pita; %pita= jumlah pita
while x<x2      %delx=lebar masing-masing pita;
   %**************************
   %fx=6-6*x^5;            %bentuk pers. 6-6x^5
   %**************************
   %fx=fabs(x); %contoh : pers. garis y=x
   %fx=1/x ;    %contoh : pers. flux magnet pada kawat berarus
```

```
    fx=x^2;
    if x==x1 || x==x2
       trap=trap+fx;
    else
       trap=trap+2*fx;
    end
    x=x+delx;
end
trap=trap*delx/2;
set(handles.hasil_trapesium_edit,'String',num2str(trap));
%Tampilkan batas integral pada sumbu
axes(handles.axes1);
x=0:0.1:100;
y=x.^2;
plot(x,y);
title('y=x^2');
xlabel('x');
ylabel('y');
for a=x1:0.01:x2  %x1=batas bawah, x2=batas atas
    Ya=a.^2;
    line([a a],[min(y) Ya],'color','r');
end
```

3.2.4 Hasil Eksekusi (*RUN*) program

Hasil eksekusi program Integral Trapesium dapat dilihat pada gambar 3.5.

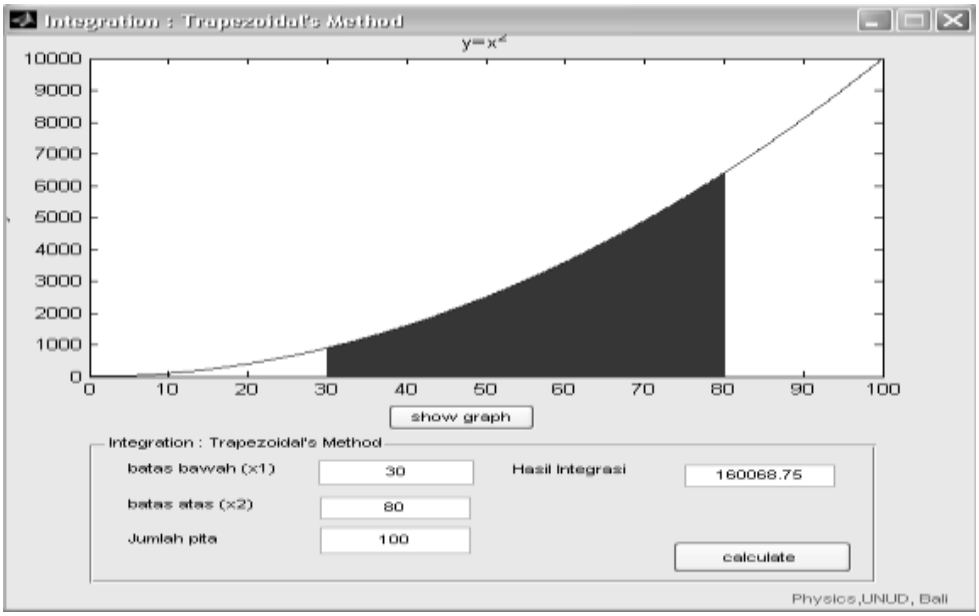


Gambar 3.5 Hasil Eksekusi program Integral Trapesium

Cara kerja program :

- Klik tombol *show graph*
- Masukkan *batas bawah (x1),* misal : 30
- Masukkan *batas atas (x2),* misal : 60
- Masukkan *Jumlah pita,* misal : 100

Maka akan didapatkan hasil seperti diperlihatkan dalam gambar 3.6.



Gambar 3.6 Hasil Akhir eksekusi program Integral Trapesium

Penjelasan hasil program :

-program menampilkan kurva dari $y = x^2$

-batas bawah(x1) : 30

Memberikan batas bawah integral yaitu 30

-batas atas (x2) : 80

Memberikan batas atas integral yaitu 80

-jumlah pita : 100

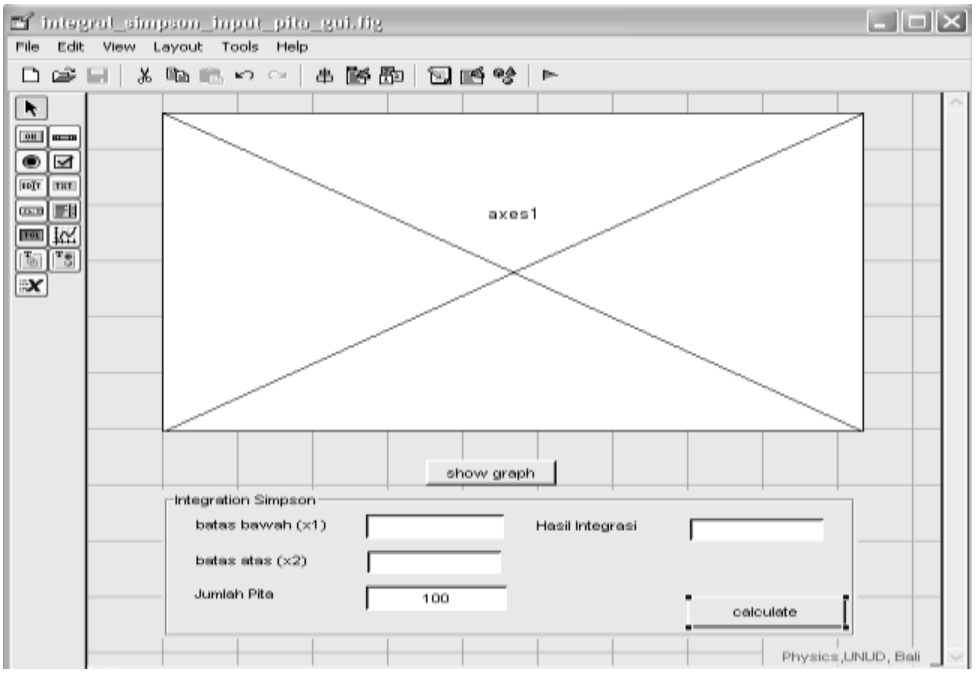
Antara batas bawah(x1) dan batas atas (x2) dibagi menjadi 100 bagian yang masing-masing berbentuk trapesium

-Hasil integrasi : 160068.75

Hasil perhitungan integral dengan metode Trapesium adalah 8865.45 yang ditunjukkan dengan *luas daerah* yang diarsir pada kurva gambar 3.6.

3.3 Program Komputer Integral : Metode Simpson

3.3.1.Rancangan Desain



Gambar 3.7 Desain GUI : Integral Simpson

3.3.2 Daftar Komponen

No	Komponen	Properti	
1	Axes1	Tag	axes1
2	Static text1	String	batas bawah (x1)
3	Static text2	String	batas atas (x2)
4	Static text3	String	Jumlah pita
5	Static text3	String	Jumlah pita
6	Edit Text1	String	
		Tag	batas_bawah_edit
7	Edit Text2	String	
		Tag	batas_atas_edit
8	Edit Text3	String	
		Tag	jml_pita_edit
9	Edit Text4	String	
		Tag	hasil_trapesium_edit
10	Pushbutton1	String	show graph
		Tag	show_pushbutton
11	Pushbutton2	String	calculate
		Tag	calculate_pushbutton
12	Panel1	Title	Integration Simpson

3.3.3. Kode Program (*source code*)

3.3.3.1 Function *show_graph_pushbutton_Callback*

```
function show_graph_pushbutton_Callback(hObject, eventdata, handles)
axes(handles.axes1);
x=0:100;
y=x.^2;
plot(x,y);
```

```
title('y=x^2');
xlabel('x');
ylabel('y');
```

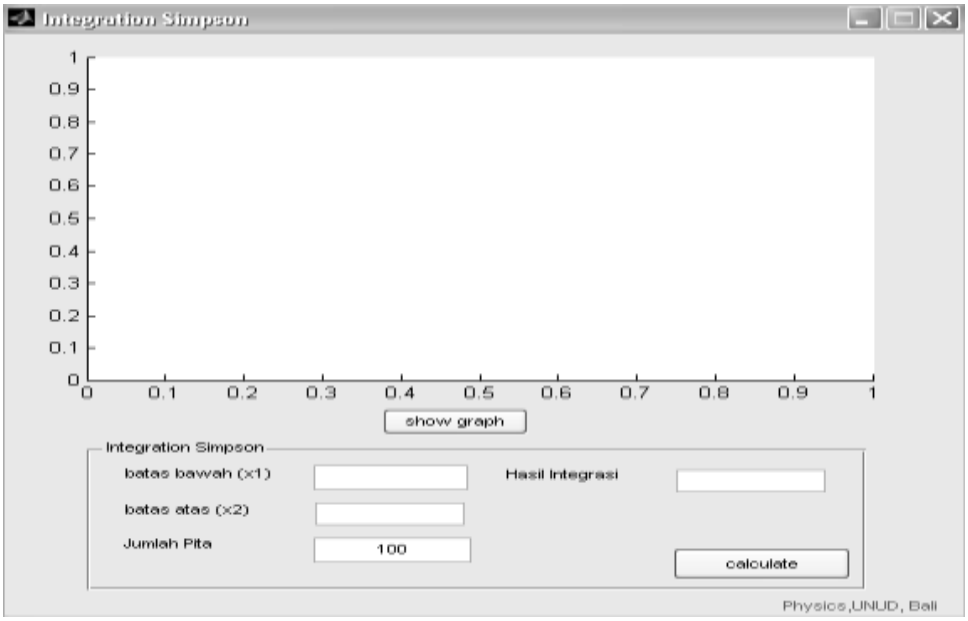
3.3.3.2 Function : *calculate_pushbutton_Callback*

```
function calculate_pushbutton_Callback(hObject, eventdata, handles)
j=0;
simp=0;
x=x1;
pita=x3;
delx=(x2-x1)/pita; %pita= jumlah pita
while x<x2        %delx=lebar masing-masing pita;
   %***************************
   %fx=6-6*pow(x,5);            %contoh : bentuk pers. 6-6x^5
   %***************************
   %fx=fabs(x); %contoh : pers. garis y=x
   %fx=1/x ;    %contoh : pers. flux magnet pada kawat berarus
   %fx=5;
   fx=x^2;
   j=j+1;
   if x==x1 || x==x2
      simp=simp+fx;
   else
      if x>x1 && x<x2 && mod(j,2) == 0  %suku genap
        simp=simp+4*fx;
      end
      if x>x1 && x<x2 && mod(j,2) ==1  %suku ganjil
        simp=simp+2*fx;
      end
   end
   x=x+delx;
end
simp=simp*delx/3;
set(handles.hasil_simpson_edit,'String',num2str(simp));
%set(handles.hasil_trapesium_edit,'String',num2str(trap(i)));
%Tampilkan batas integral pada sumbu
axes(handles.axes1);
plot(handles.x,handles.y);
title('y=x^2');
xlabel('x');
ylabel('y');
%showing shaded area under curve between x1 and x2
```

```
for a=x1:0.01:x2  %x1=batas bawah, x2=batas atas
   Ya=a.^2;
   line([a a],[min(handles.y) Ya],'color','g');
end
```

3.2.4 Hasil Eksekusi (*RUN*) program

Hasil eksekusi program Integral Simpson dapat dilihat pada Gambar 3.8.

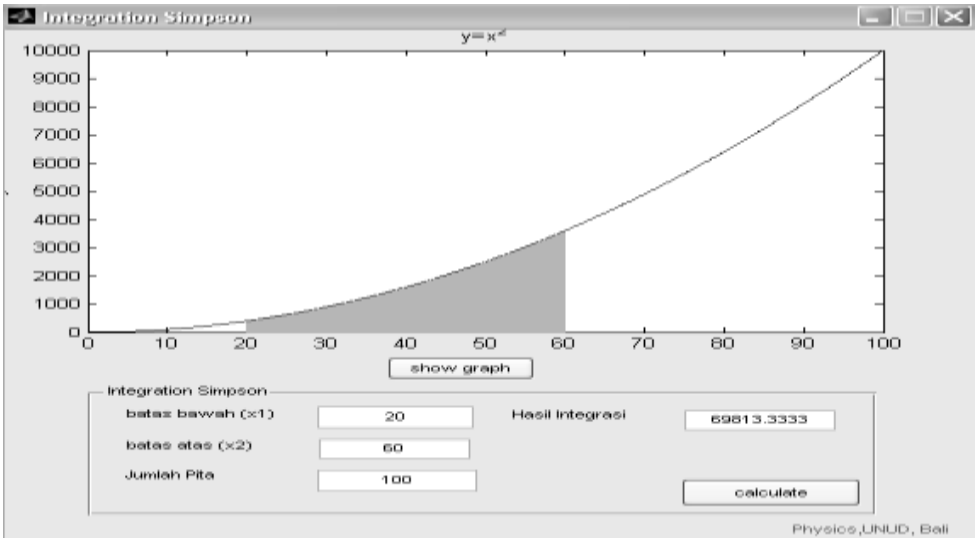


Gambar 3.8 Hasil Eksekusi program Integral Simpson

Cara kerja program :

- Klik tombol *show graph*
- Masukkan *batas bawah (x1)*, misal : 20
- Masukkan *batas atas (x2)*, misal : 60
- Masukkan *Jumlah pita*, misal : 100

Maka akan didapatkan hasil seperti diperlihatkan dalam gambar 3.9.



Gambar 3.9 Hasil Akhir eksekusi program Integral Simpson

Penjelasan hasil program :

-program menampilkan kurva dari $y = x^2$

-batas bawah(x1) : 20

Memberikan batas bawah integral yaitu 20

-batas atas (x2) : 60

Memberikan batas atas integral yaitu 60

-jumlah pita : 100

Antara batas bawah(x1) dan batas atas (x2) dibagi menjadi 100 bagian yang masing-masing berbentuk polinom orde 2

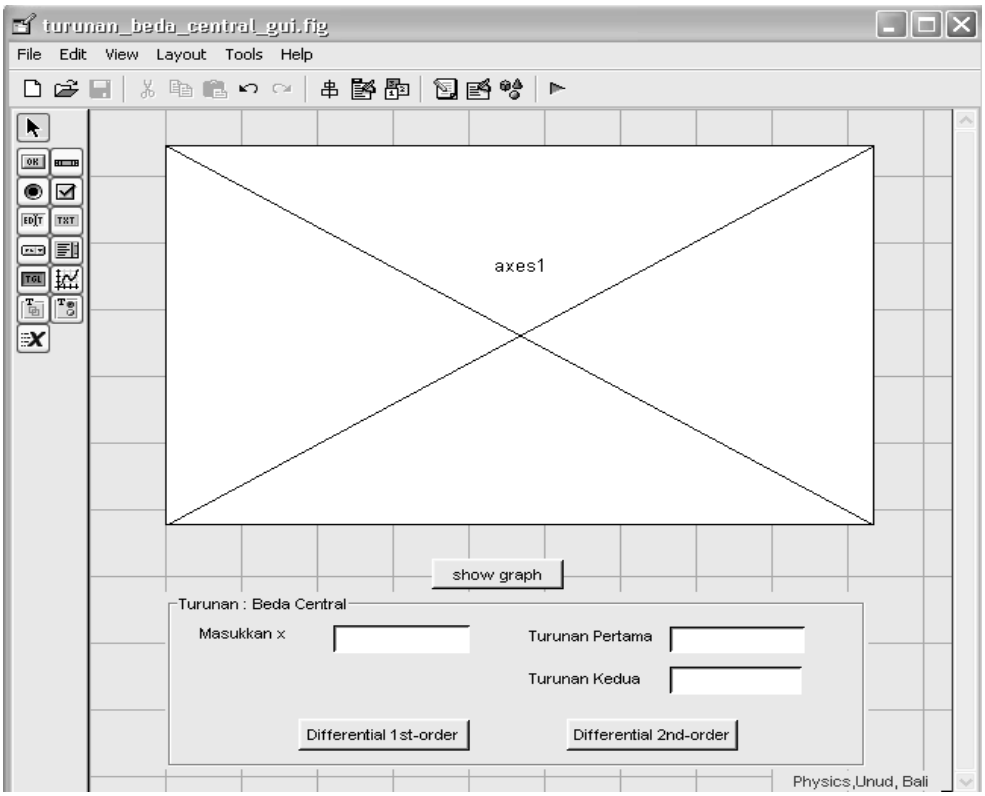
-Hasil integrasi : 69813.3333

Hasil perhitungan integral dengan metode Simpson adalah 69813.3333 yang ditunjukkan dengan *luas daerah* yang diarsir pada kurva gambar 3.9.

4 DIFERENSIAL

4.1 Program Komputer Diferensial : Metode Beda Central (*Central Difference*)

4.1.1.Rancangan Desain



Gambar 4.1 Desain GUI Diferensial Beda Central

4.1.2 Daftar Komponen

No	Komponen	Properti	
1	Axes1	Tag	axes1
2	Static text1	String	Masukkan x
3	Static text2	String	Turunan Pertama
4	Static text3	String	Turunan Kedua
5	Edit Text1	String	
		Tag	x_edit
6	Edit Text2	String	
		Tag	hasil_turunan_pertama_edit
7	Edit Text3	String	
		Tag	Hasil_turunan_kedua_edit
8	Pushbutton1	String	show graph
		Tag	show_pushbutton
9	Pushbutton2	String	Differential 1st-order
		Tag	calculate_differential1_pushbutton
10	Pushbutton3	String	Differential 2nd-order
		Tag	calculate_differential2_pushbutton
12	Panel1	Title	Turunan : Beda Central

4.1.3. Kode Program (*source code*)

4.1.1.1 Function *show_graph_pushbutton_Callback*

```
function show_graph_pushbutton_Callback(hObject, eventdata, handles)
axes(handles.axes1);
x=0:0.1:100;
y=x.^2-5.*x;
plot(x,y);
title('grafik y=x^2-5x');
xlabel('x');
ylabel('y');
```

4.1.1.2 Function : *calculate_differential1_pushbutton_Callback*

```
function calculate_differential1_pushbutton_Callback(hObject, eventdata, handles)
y = str2num(get(handles.x_edit,'String'));
eps=1e-5;
h=1;
%n=0;
zz=0;
del=10;
while abs(del)>=eps %iterasi selama del lebih besar dari eps
  % n=n+1;
   for i=1:1:2
      if i==1
         x=y+h;
      else
         x=y-h;
      end
      %fx=9.8*68.1*(1-exp(-12.5*x/68.1));
      %fx=fx/12.5;
      %*****************
      fx=x*x-5*x;         %fungsi yang dicari turunannya : x^2 - 5x
      %*****************
      z(i)=fx;
   end
   dx=(z(1)-z(2))/(2*h);
   del=zz-dx;
   zz=dx;
   h=h/10;
```

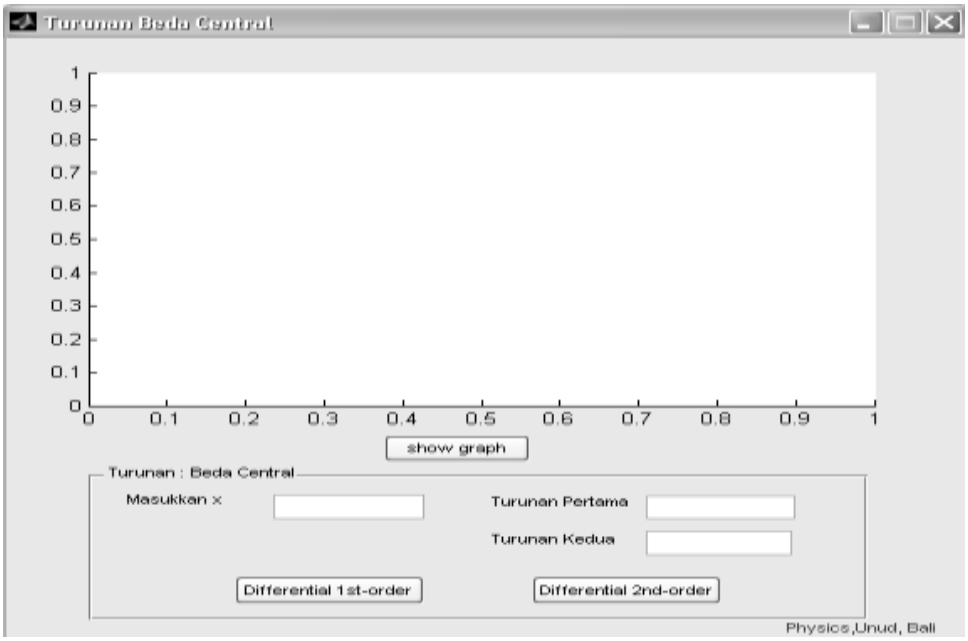
```
end
set(handles.hasil_turunan_pertama_edit,'String',num2str(dx));
```

4.1.1.3 Function : *calculate_differential2_pushbutton_Callback*

```
function calculate_differential2_pushbutton_Callback(hObject, eventdata, handles)
y = str2num(get(handles.x_edit,'String'));
eps=1e-5;
h=1;
n=0;
zz=0;
del=10;
while abs(del)>=eps %iterasi selama del lebih besar dari eps
   n=n+1;
   for i=1:1:3
      switch i
         case 1
            x=y+h;
         case 2
            x=y;
         case 3
            x=y-h
      end
      %fx=9.8*68.1*(1-exp(-12.5*x/68.1));
      %fx=fx/12.5;
      %*****************
      fx=x*x-5*x;          %fungsi yang dicari turunannya : x^2 - 5x
      %*****************
      z(i)=fx;
   end
       dx=(z(1)-2*z(2)+z(3))/(h*h);
   del=zz-dx;
   zz=dx;
   h=h/10;
end
set(handles.hasil_turunan_kedua_edit,'String',num2str(dx));
```

4.1.4 Hasil Eksekusi (*RUN*) program

Hasil eksekusi program Beda Central dapat dilihat pada gambar 3.11.

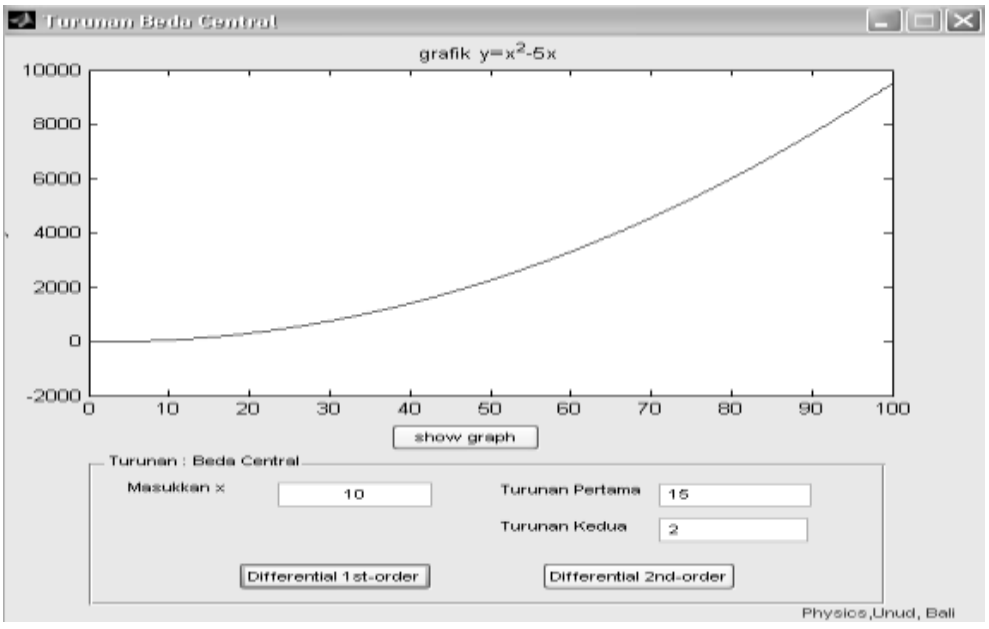


Gambar 3.11 Hasil Eksekusi program Beda Central

Cara kerja program :

- Klik tombol *show graph*
- Masukkan x, misal : 10
- Klik tombol *Differential 1st -order*
- Klik tombol *Differential 2nd –order*

Maka akan didapatkan hasil seperti diperlihatkan dalam gambar 3.12.



Gambar 3.12 Hasil Akhir eksekusi program Beda Central

Penjelasan hasil program :

-program menampilkan kurva dari $y = x^2 - 5x$

-masukkan x : 10

Menyatakan akan dihitung turunan saat x = 10 atau $\left.\frac{dy}{dx}\right|_{x=10}$

-Turuan Pertama : 15

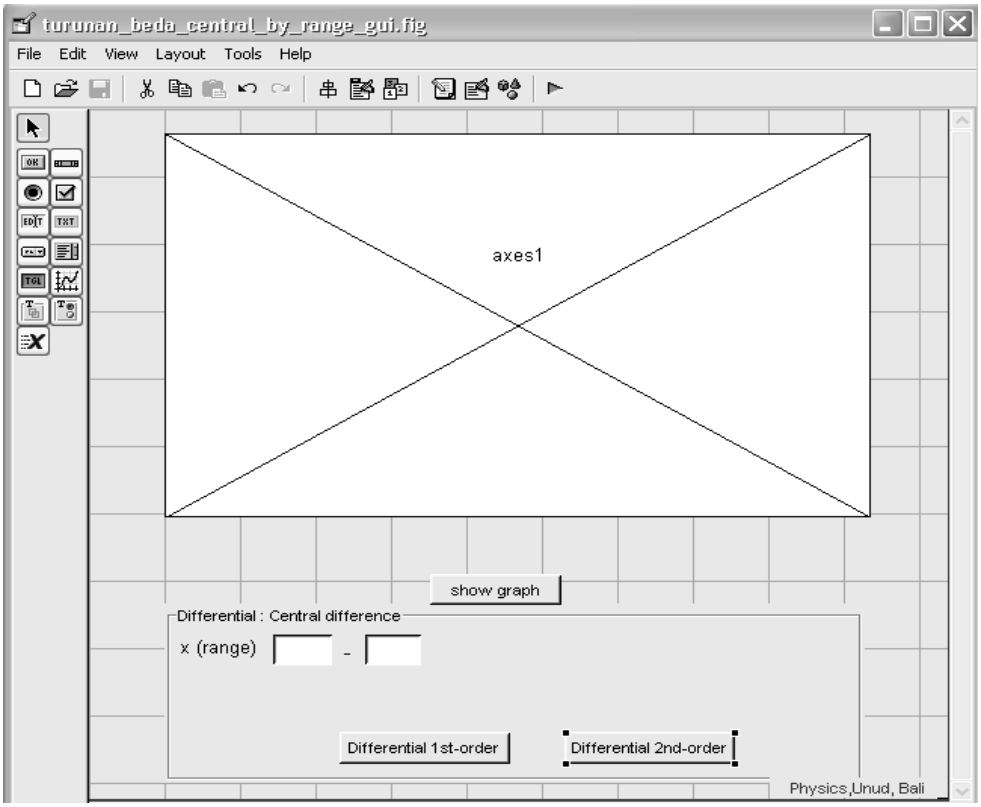
Hasil turunan pertama $\left.\frac{dy}{dx}\right|_{x=10} = 15$

-Turunan Kedua : 2

Hasil turunan kedua $\left.\frac{d^2y}{dx^2}\right|_{x=10} = 2$

4.2 Program Komputer Diferensial : Metode Beda Central (dengan range x)

4.2.1.Rancangan Desain



Gambar 3.13 Desain GUI : Beda Central (range x)

4.2.2 Daftar Komponen

No	Komponen	Properti	
1	Axes1	Tag	axes1
2	Static text1	String	x (range)
3	Edit Text1	String	
		Tag	x_awal_edit
4	Edit Text2	String	
		Tag	X_akhir_edit
5	Pushbutton1	String	show graph
		Tag	show_graph_pushbutton
6	Pushbutton2	String	Differential 1st-order
		Tag	calculate_differential1_pushbutton
7	Pushbutton3	String	Differential 2nd-order
		Tag	calculate_differential2_pushbutton
8	Panel1	Title	Differential : Central Difference

4.2.3. Kode Program (*source code*)

4.2.3.1 Function *show_graph_pushbutton_Callback*

```
axes(handles.axes1);
handles.x=0:(2*pi)/100:2*pi;
handles.y=sin(handles.x);
handles.p=plot(handles.x,handles.y);
title('y=sin(x)');
xlabel('x');
ylabel('y');
guidata(hObject,handles);
```

4.2.3.2 Function : *calculate_differential1_pushbutton_Callback*

```
x_1=str2num(get(handles.x_awal_edit,'String'));
x_2=str2num(get(handles.x_akhir_edit,'String'));
jml_pita = 100;
interval_x =(x_2-x_1)/jml_pita;
n=1;
y=x_1;
while str2num(num2str(y,'%.2f'))<=str2num(num2str(x_2,'%.2f')) %yhanya sampai 2
desimal
   %y = str2num(get(handles.x_edit,'String'));
   eps=1e-5;
   h=1;
   %n=0;
   zz=0;
   del=10;
   while abs(del)>=eps %iterasi selama del lebih besar dari eps
      for i=1:1:2
         if i==1
            x=y+h;
```

```
            else
               x=y-h;
            end
            %fx=9.8*68.1*(1-exp(-12.5*x/68.1));
            %fx=fx/12.5;
            %*****************
            %fx=x*x-5*x;         %fungsi yang dicari turunannya : x^2 - 5x
            fx=sin(x);
            %*****************
            z(i)=fx;
         end
         dx=(z(1)-z(2))/(2*h); %rumus turunan beda central
         del=zz-dx;
         zz=dx;
         h=h/10;
      end
      dif(n)=dx;
      xn(n)=y;
      y=y+interval_x;
      n=n+1;
   end
   %************************************************
   %tampilkan turunan pertama di sumbu yg sama (axis1)
   %************************************************
   handles.y(2,:)=dif; %set baris kedua dari y sbg turunan pertama
   %set(handles.p,plot(x,handles.y(2,:)));
   plot(handles.x,handles.y);
   %plot(xn,dif,'+');
   title('y=sin(x)');
   xlabel('x');
   ylabel('y');
   guidata(hObject,handles);
```

4.2.3.3 Function : *calculate_differential2_pushbutton_Callback*

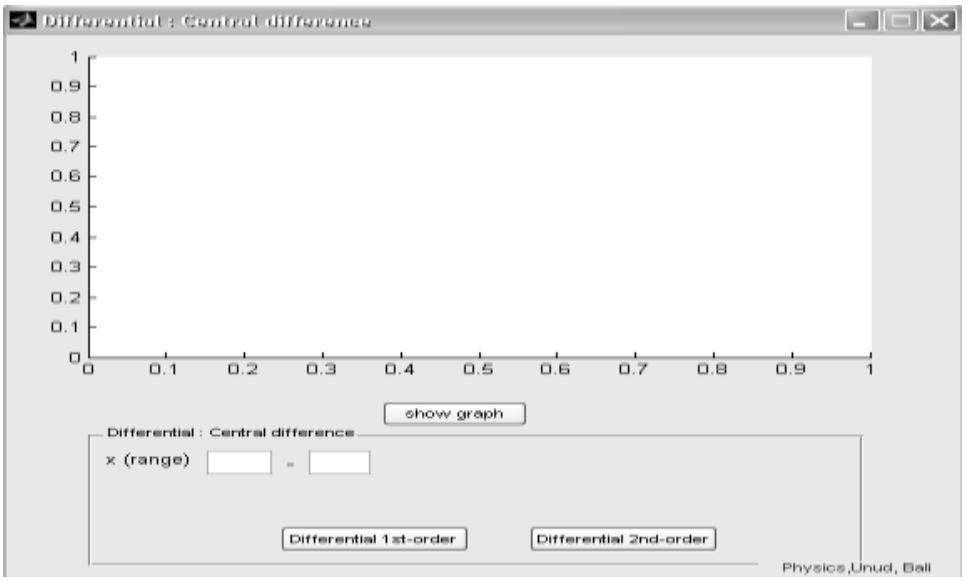
```
   x_1=str2num(get(handles.x_awal_edit,'String'));
   x_2=str2num(get(handles.x_akhir_edit,'String'));
   jml_pita = 100;
   interval_x =(x_2-x_1)/jml_pita;
   n=1;
   y=x_1;
   while str2num(num2str(y,'%.2f'))<=str2num(num2str(x_2,'%.2f')) %y hanya sampai 2
   desimal
```

```
    %y = str2num(get(handles.x_edit,'String'));
    eps=1e-5;
    h=1;
    %n=0;
    zz=0;
    del=10;
    while abs(del)>=eps %iterasi selama del lebih besar dari eps
      %n=n+1;
      for i=1:1:3
        switch i
          case 1
            x=y+h;
          case 2
            x=y;
          case 3
            x=y-h;
        end
        %fx=9.8*68.1*(1-exp(-12.5*x/68.1));
        %fx=fx/12.5;
        %*****************
        fx=sin(x);        %fungsi yang dicari turunannya
        %*****************
        z(i)=fx;
      end
      dx=(z(1)-2*z(2)+z(3))/(h*h); %Hitung turunan kedua
      del=zz-dx;
      zz=dx;
      h=h/10;
    end
    dif2(n)=dx;
    xn(n)=y;
    y=y+interval_x;
    n=n+1;
end
handles.y(3,:)=dif2; %set baris ketiga dari y sbg turunan kedua
%set(handles.p,plot(x,handles.y(2,:)));
plot(handles.x,handles.y);
%plot(handles.x,handles.y(3,:));
%plot(xn,dif,'+');
title('y=sin(x)');
xlabel('x');
```

```
ylabel('y');
guidata(hObject,handles);
```

4.2.4 Hasil Eksekusi (*RUN*) program

Hasil program Beda Central (x range) dilihat pada gambar 3.14.

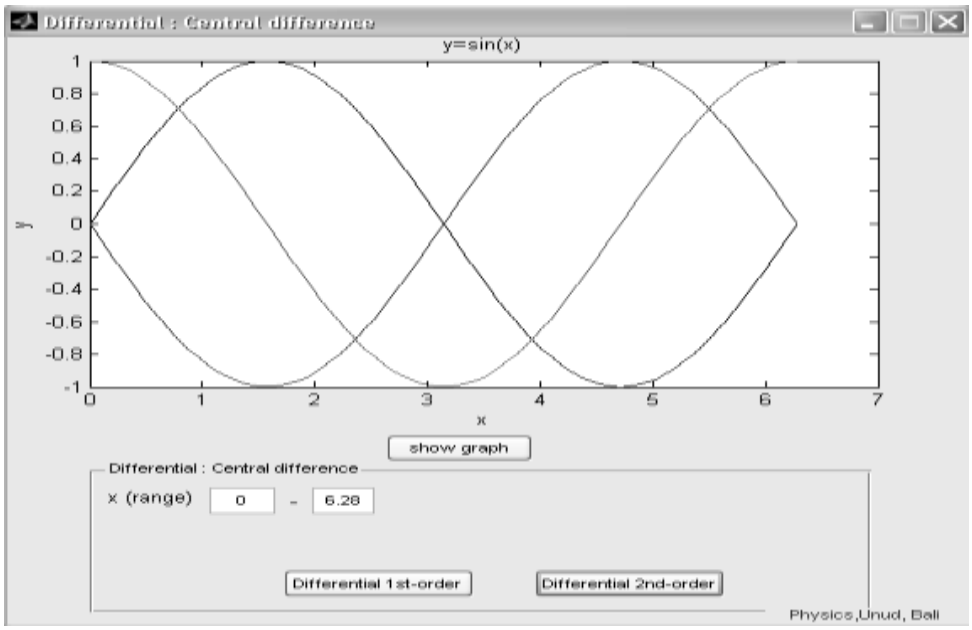


Gambar 3.14 Hasil Eksekusi program Beda Central (x range)

Cara kerja program :

- Klik tombol *show graph*
- Masukkan x (range) , misal : 0 6.28
- Klik tombol *Differential 1st -order*
- Klik tombol *Differential 2nd -order*

Maka akan didapatkan hasil seperti diperlihatkan dalam gambar 3.15.



Gambar 3.15 Hasil Akhir eksekusi program Beda Central (x range)

Penjelasan hasil program :

- program menampilkan kurva dari y = sin(x), dinyatakan dengan kurva berwarna biru
- masukkan x : 0 6.28

  Menyatakan akan dihitung turunan dari x = 0 sampai x = 6.28
- Hasilnya adalah :

  Hasil Turunan pertama dinyatakan dengan grafik kurva berwarna *hijau*, yang mana bentuknya seperti kurva fungsi cos(x). Sedangkan hasil turunan kedua dinyatakan dengan kurva berwarna *merah*, dengan bentuk kurva seperti kurva fungsi –sin(x)

DAFTAR PUSTAKA

Abdul Munif, Aries Prastyoko H., Cara *Praktis Penguasaan dan Penggunaan Metode Numerik*, Penerbit Guna Widya, Surabaya, 1995

Ahmad Basuki, Nana Ramadijanti, *Metode Numerik dan Algoritma Komputasi*, Penerbit Andi, Jogyakarta, 2005

Chapra Steven C., Canale Raymond P., *Numerical Methods for Engineers*, Mc Graw-Hill Book Company, New York, 1985

De Vries, P.L., *A First Course in Computational Physics*, John Wiley & Sons, USA,1999

R.Soegeng, *Komputasi Numerik dengan Turbo Pascal*, Penerbit Andi, Jogyakarta, 1993

Young, M.J. *Mastering Visual C++*, Sibex, 1998