Lecture 6III While-Looping

- Binary search
- While loop
  - Root Finding

An array of sorted integers

- X is an array of n sorted integers
- X[i] < X[j], if i < j
- X=[1 3 5 7 9 11 13 15 17]
- e.x.   X[3] = 5   < X[6] =11

Searching

- s is an integer
- Is s in X ?
- Sequential search: Compare s with X[1],X[2],…,X[n] one by one
- Time consuming

- For large n, it is time consuming to compare s with elements in X one by one
- Binary search improves this drawback
- Binary search is more efficient than sequential search

An array of sorted elements

- X is a vector of positive integers
- X[i] < X[j] if i < j
- Input s
- Output i
  - i > 0, where x[i] == s
  - i=0, if s ~= x[i] for all i

Example

- X=[1 3 5 7 9 11 13 15 17] and s=5
- Output : 3

- X=[1 3 5 7 9 11 13 15 17] and s=4
- Output : 0

Sorting

```
N=10;
X=ceil(rand(1,N)*10000);
X=sort(X)
```

```
X =

  Columns 1 through 7

      1704      1834      2141      2633      6022      6050      6366

  Columns 8 through 10

      6596      6597      7537
```

```
N=500;
X=ceil(rand(1,N)*10000);
X=sort(X)


 a=1;b=length(X);
```

- Binary search cuts [a,b] into two sub-interval

$$[a,c-1]\cup\{c\}\cup[c+1,b]$$

- Where c=floor((a+b)/2)

- Halting condition: s == X[c] | a > b
- If s < X[c], the answer should belong [a,c-1]

$$b \leftarrow c - 1$$

- If X[c] < s, the answer should belong [c+1,b]

$$a \leftarrow c + 1$$

left interval

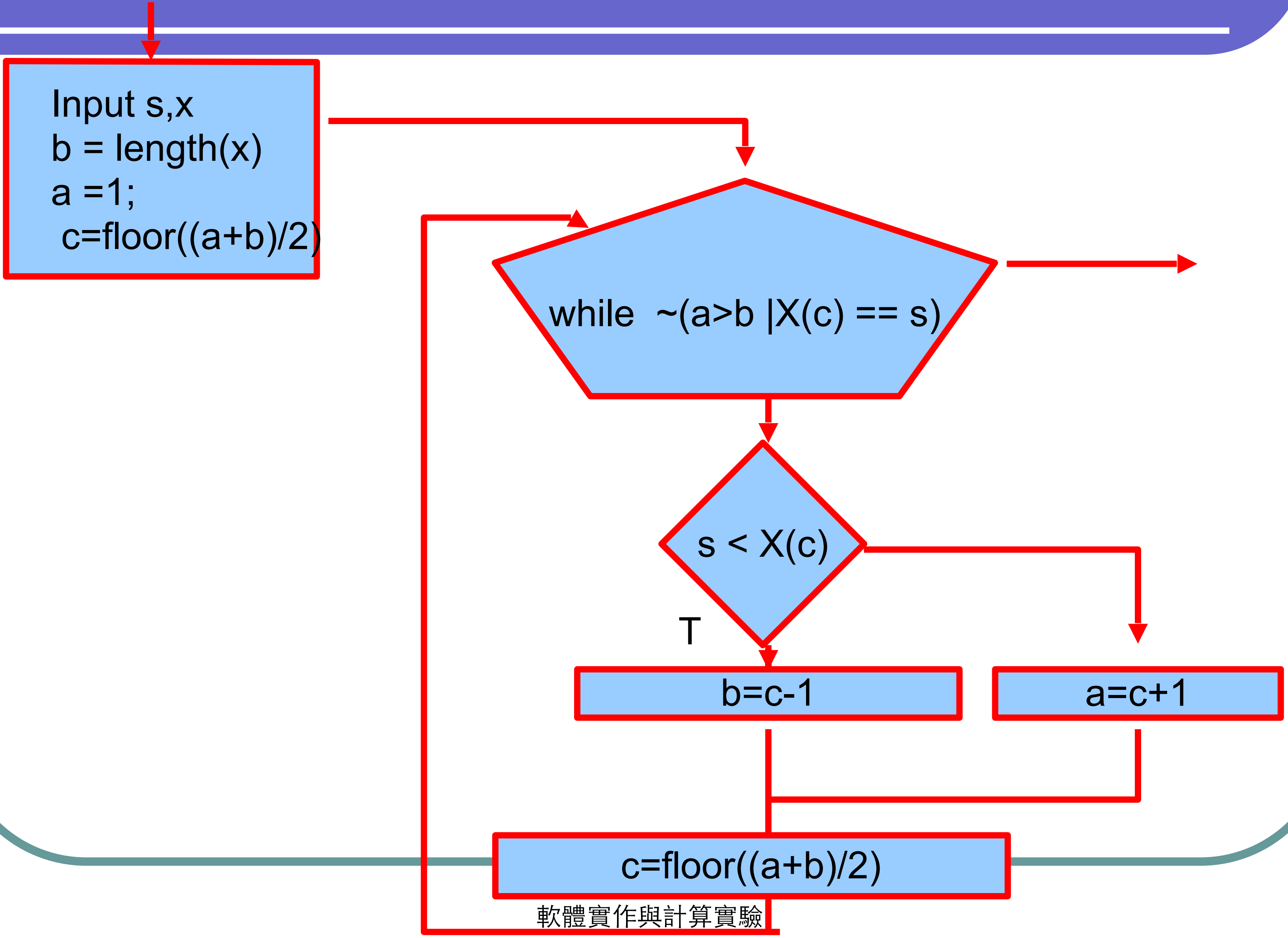
- X=[1 3 5 7 9 11 13 15 17] and s=4
- a=1; b=9
- c= 5
- The location of s is within [a,c]
- By operation, $b \leftarrow c - 1$
  the searching interval [a b] becomes [1 4]

Right interval

- X=[1 3 5 7 9 11 13 15 17] and s=13
- a=1; b=9
- c= 5
- The location of s is within [c,b]
- By operation $a \leftarrow c+1$
  the searching interval [a b] becomes [6 9]

Right interval

- X=[1 23 45 38 66 77 88 99 101 999] and s=66
- a=1; b=5
- c= 3
- The location of s is within [c,b]
- By operation $a \leftarrow c$
  the searching interval [a b] becomes [3 5]


Input s,x
b = length(x)
a =1;
c=floor((a+b)/2)
while ~(a>b |X(c) == s)
s < X(c)
T
b=c-1
a=c+1
c=floor((a+b)/2)

- Entry condition

  $\sim(a>b \mid X(c) == s)$

- Halting condition

  $(a>b \mid X(c) == s)$

Inline : f(x) = sin(x)

```
fs = input('f(x) = ','s');
f  = inline(fs);
```

```
>> f(pi/2)

ans =

     1
```

Roots

- Mathematics
  - x is a root of f(x) if f(x) = 0
- Numerical analysis
  - x is a root of f(x) if abs(f(x)) < eps

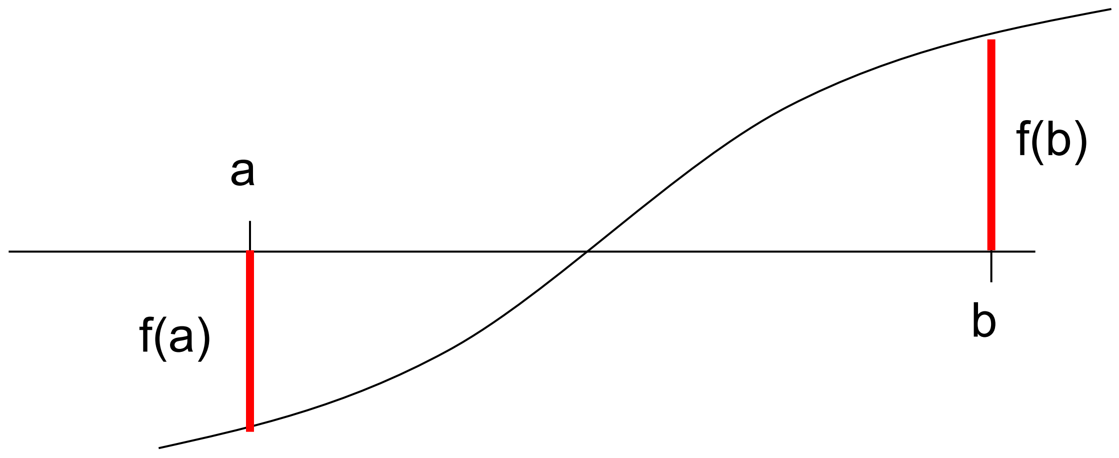
```
>> sin(pi)

ans =

  1.2246e-016
```

```
>> abs(sin(pi)) < eps

ans =

     1
```

Existence of roots

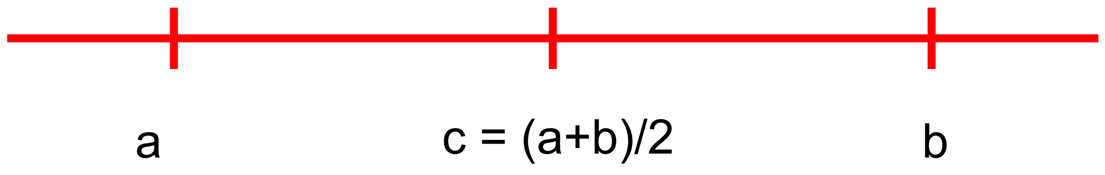
- f(x) is well defined over interval [a,b]
- If f(x) is continuous and f(a)f(b) < 0

  there exists at least one root within [a,b]



Bipartition

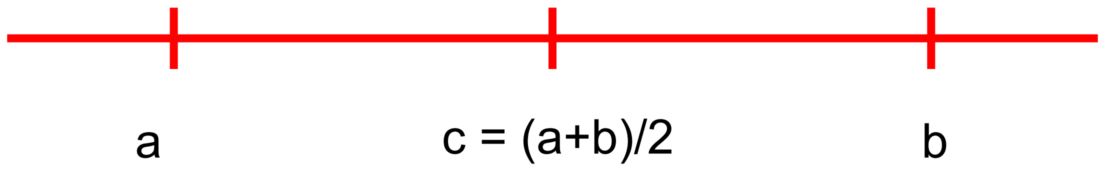
- Partition interval [a,b] to two intervals such that [a,b]=[a,c] U [c,b], where c = (a+b) / 2



Proposition

If $f(a)f(b) < 0$

it holds that $f(a)f(c) < 0$ or $f(c)f(b) < 0$



Proof

$f(a)f(b) < 0, f(c) \neq 0$

(I) $f(a) > 0$ and $f(b) < 0$

$\Rightarrow f(c)f(a) < 0$ or $f(c)f(b) < 0$

(II) $f(a) < 0$ and $f(b) > 0$

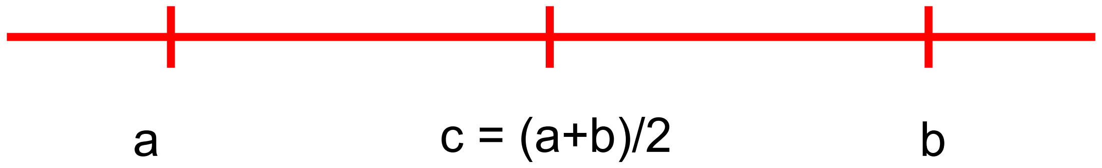
$\Rightarrow f(c)f(a) < 0$ or $f(c)f(b) < 0$

Target interval

If $f(a)f(b) < 0$

it holds that $f(a)f(c) < 0$ or $f(c)f(b) < 0$

If $f(a)f(c) < 0$, choose [a c] as target interval
If $f(c)f(b) < 0$, choose [c b] as target interval



Binary search

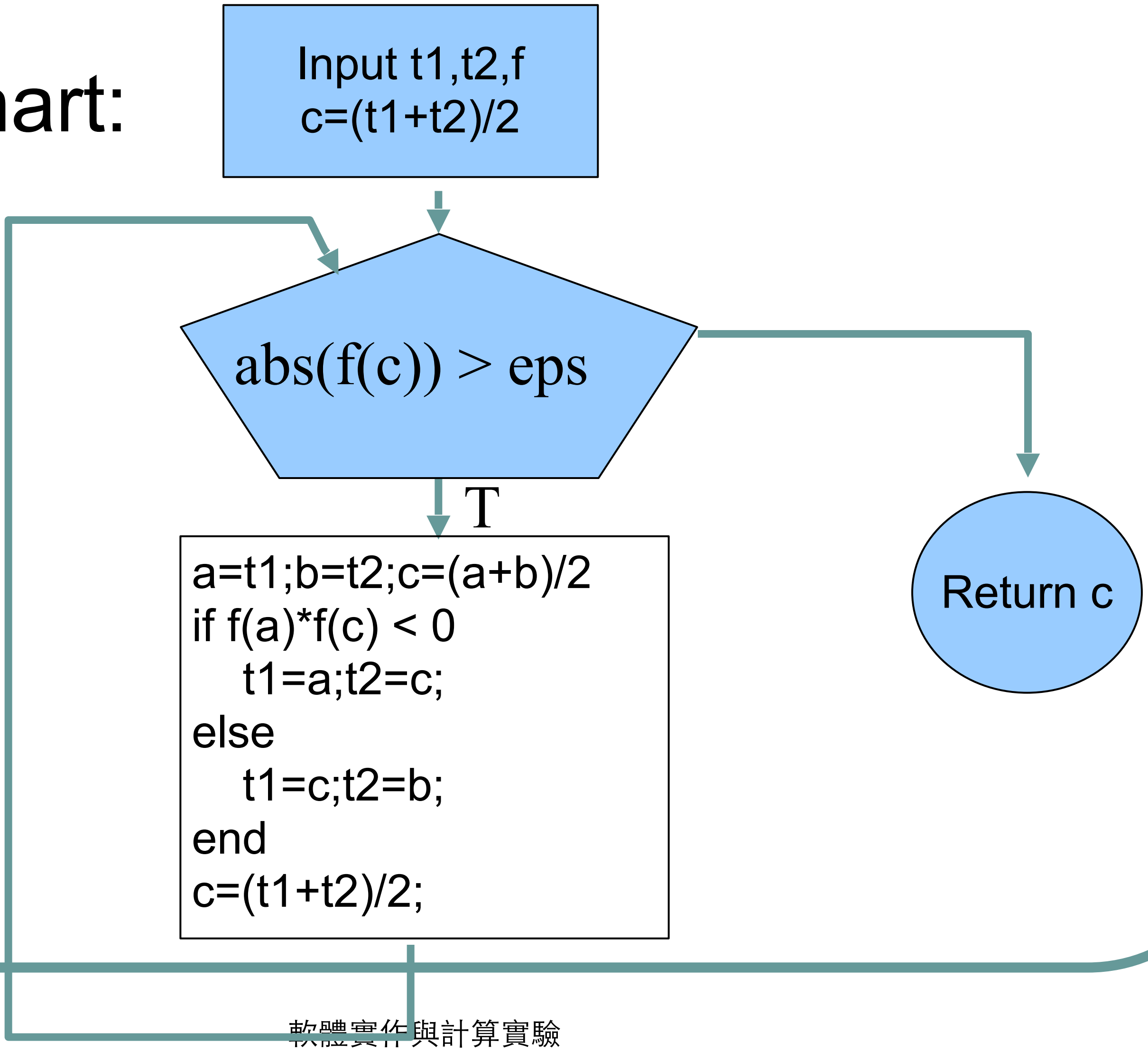
- Target interval [t1 t2]
- If halting condition holds, halt
- $a \leftarrow t1$, $b \leftarrow t2$, $c \leftarrow (t1+t2)/2$
- If $f(a)f(c) < 0$, $t1 \leftarrow a$, $t2 \leftarrow c$ choose [a c]
- If $f(c)f(b) < 0$, $t1 \leftarrow c$, $t2 \leftarrow b$ choose [c b]

Halting condition

- c=(t1+t2)/2
- f(c) is close enough to zero
- Implementation
  - abs(f(c)) < eps

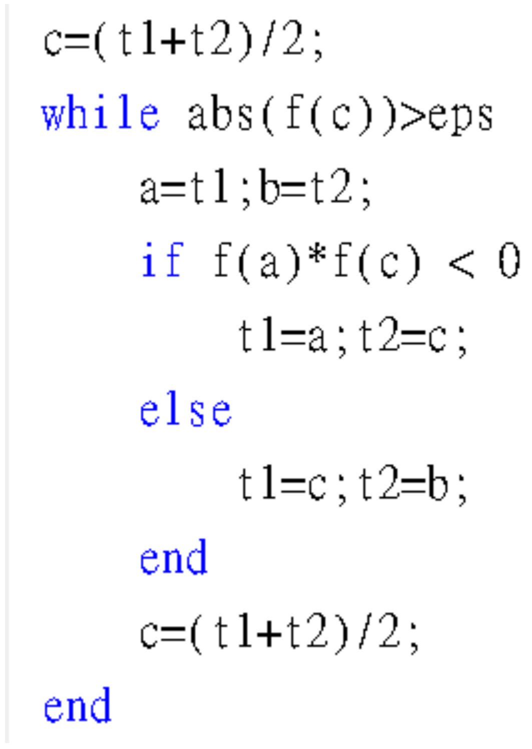
Zero finding

- Flow Chart:



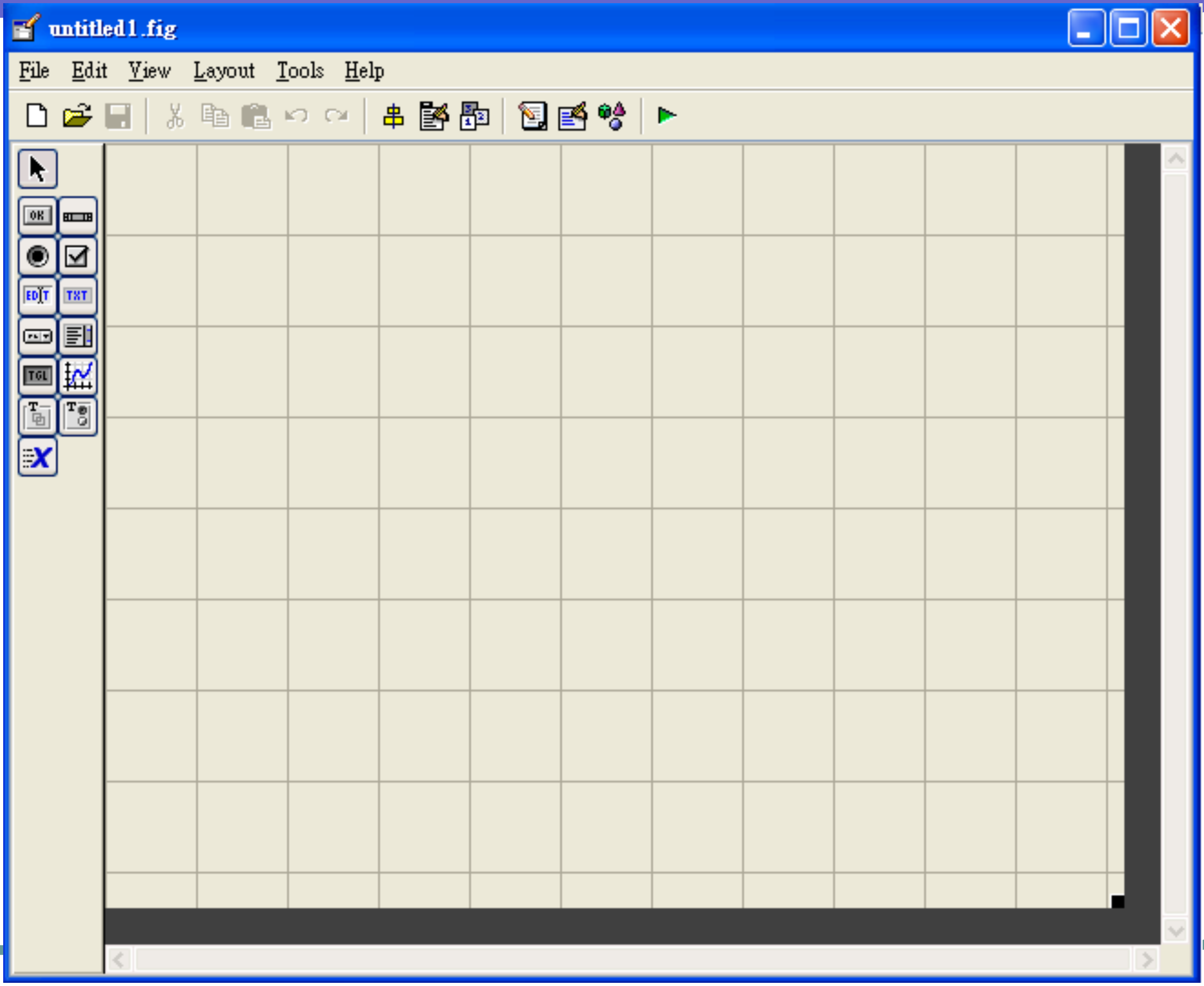
```
a=t1;b=t2;c=(a+b)/2
if f(a)*f(c) < 0
    t1=a;t2=c;
else
    t1=c;t2=b;
end
c=(t1+t2)/2;
```

my_fzero.m

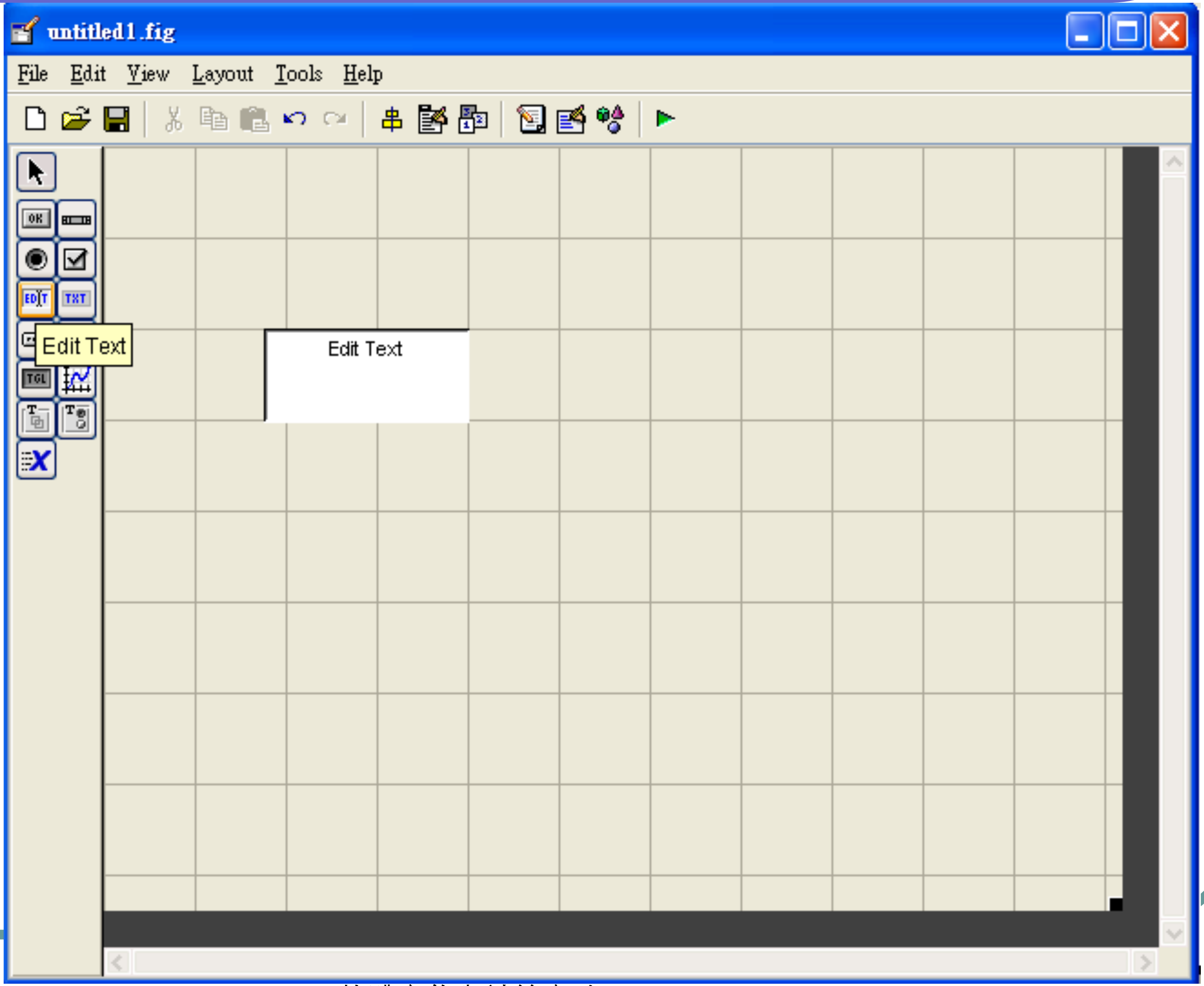


```
c=(t1+t2)/2;
while abs(f(c))>eps
    a=t1;b=t2;
    if f(a)*f(c) < 0
        t1=a;t2=c;
    else
        t1=c;t2=b;
    end
    c=(t1+t2)/2;
end
```

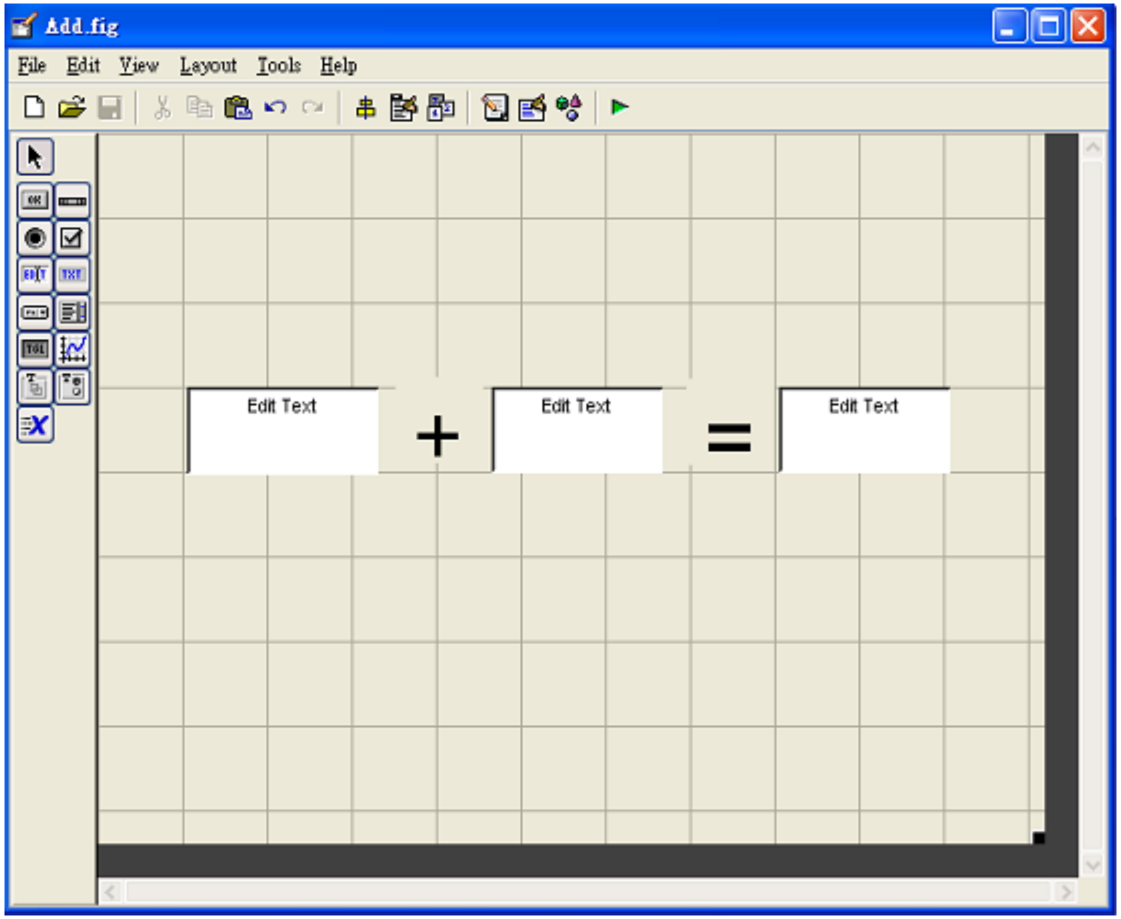
New

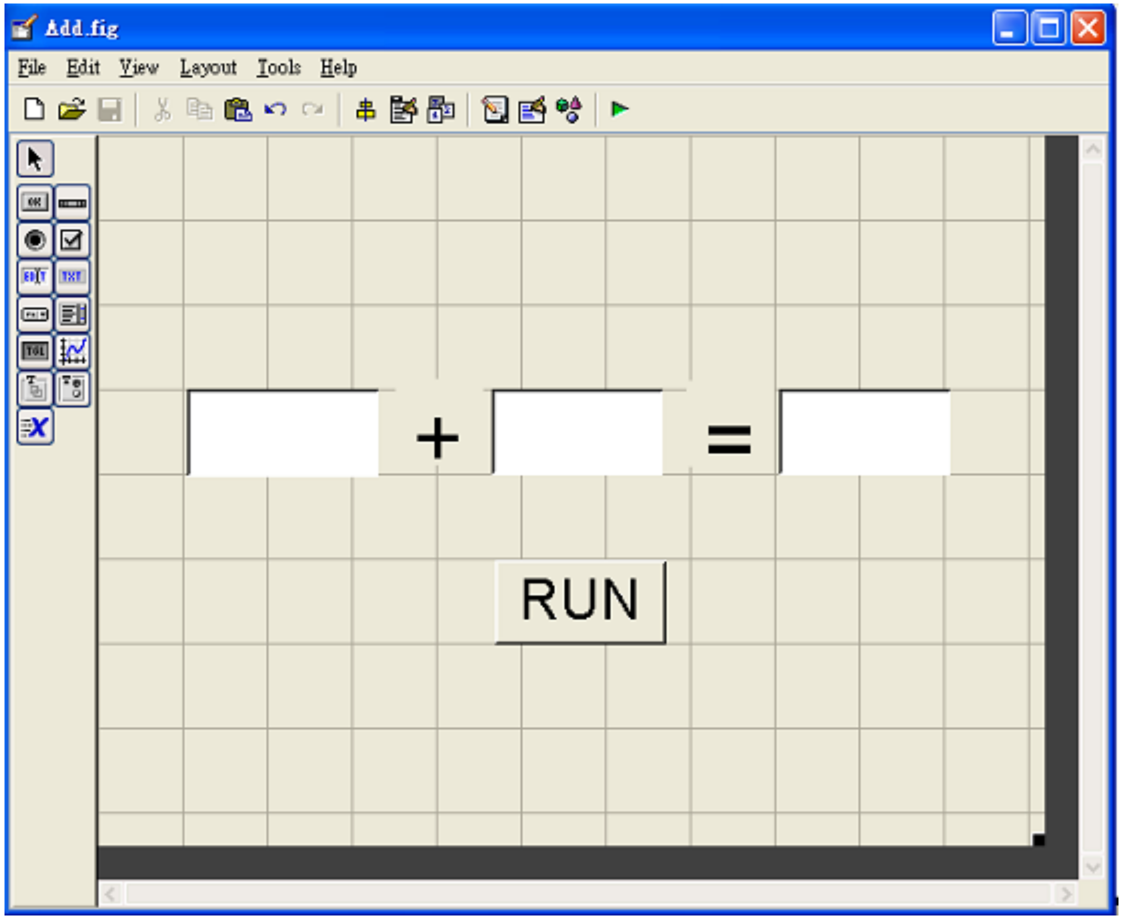

untitled1.fig
File Edit View Layout Tools Help
OK
EDIT
TXT
TGL

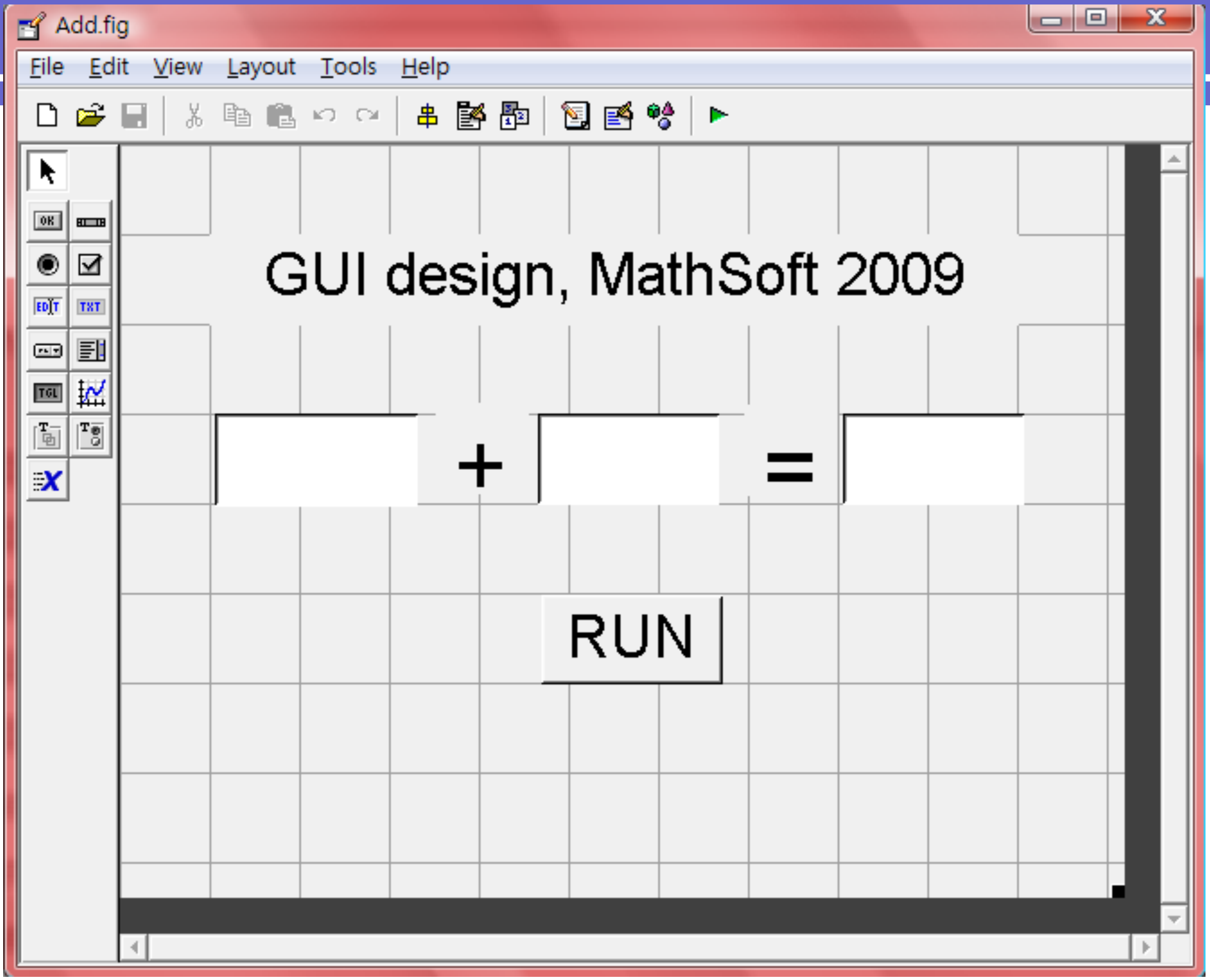
Edit Text


untitled1.fig
File
Edit
View
Layout
Tools
Help
Edit Text
Edit Text

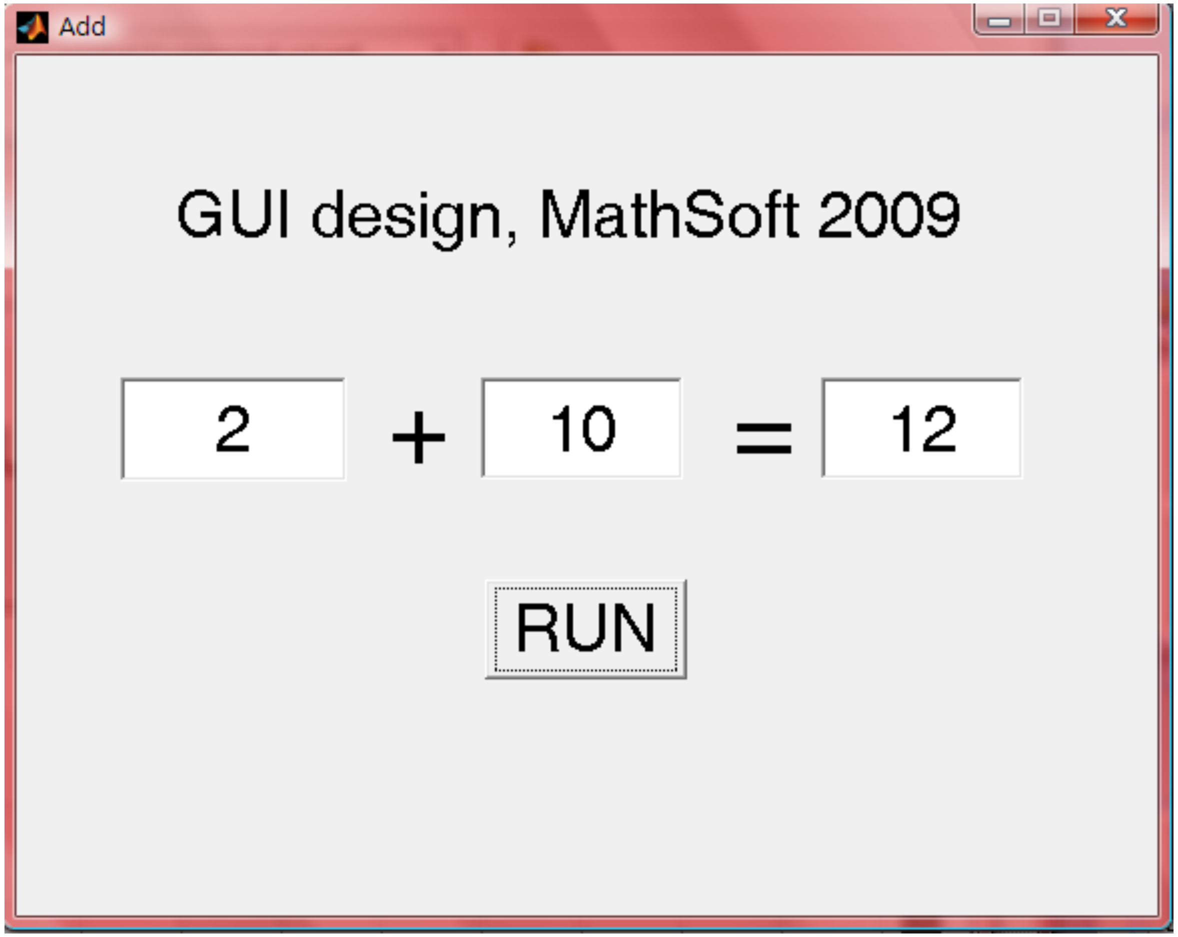
Addition


Add.fig
File Edit View Layout Tools Help
Edit Text
+
Edit Text
=
Edit Text


Add.fig
File Edit View Layout Tools Help
+
=
RUN


Add.fig
File
Edit
View
Layout
Tools
Help
GUI design, MathSoft 2009
+
=
RUN

```
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
s1=get(handles.edit1,'String');
s2=get(handles.edit2,'String');
x=str2double(s1)+str2double(s2);
s3=num2str(x);
set(handles.edit3,'String',s3);
return
```


Add
GUI design, MathSoft 2009
2
+
10
=
12
RUN

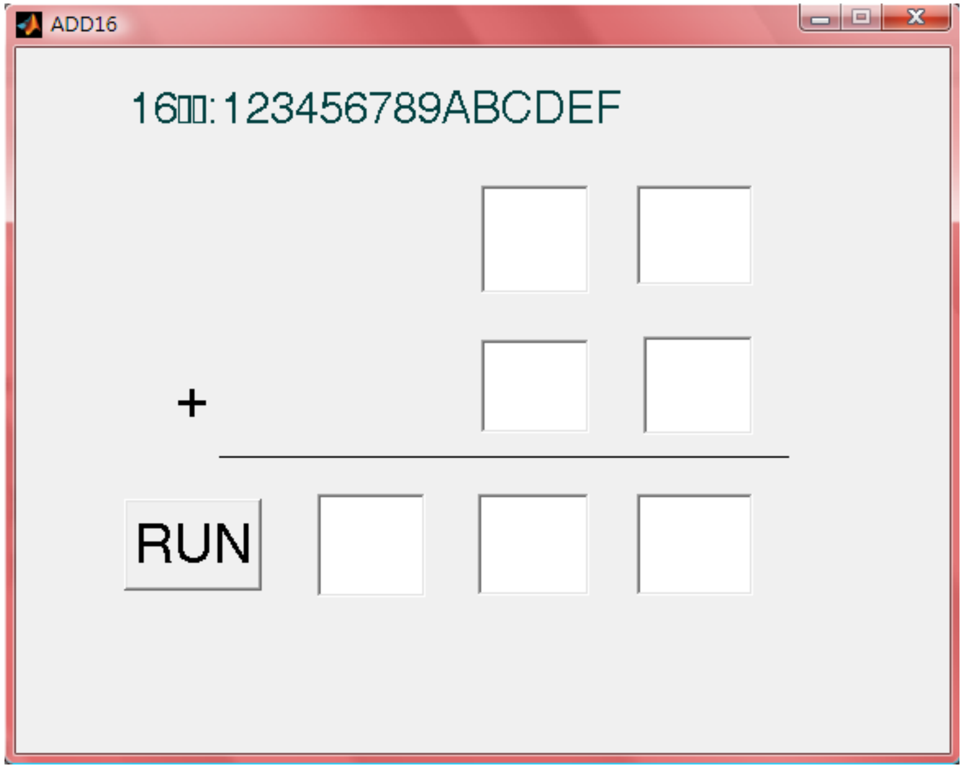
GUI tool

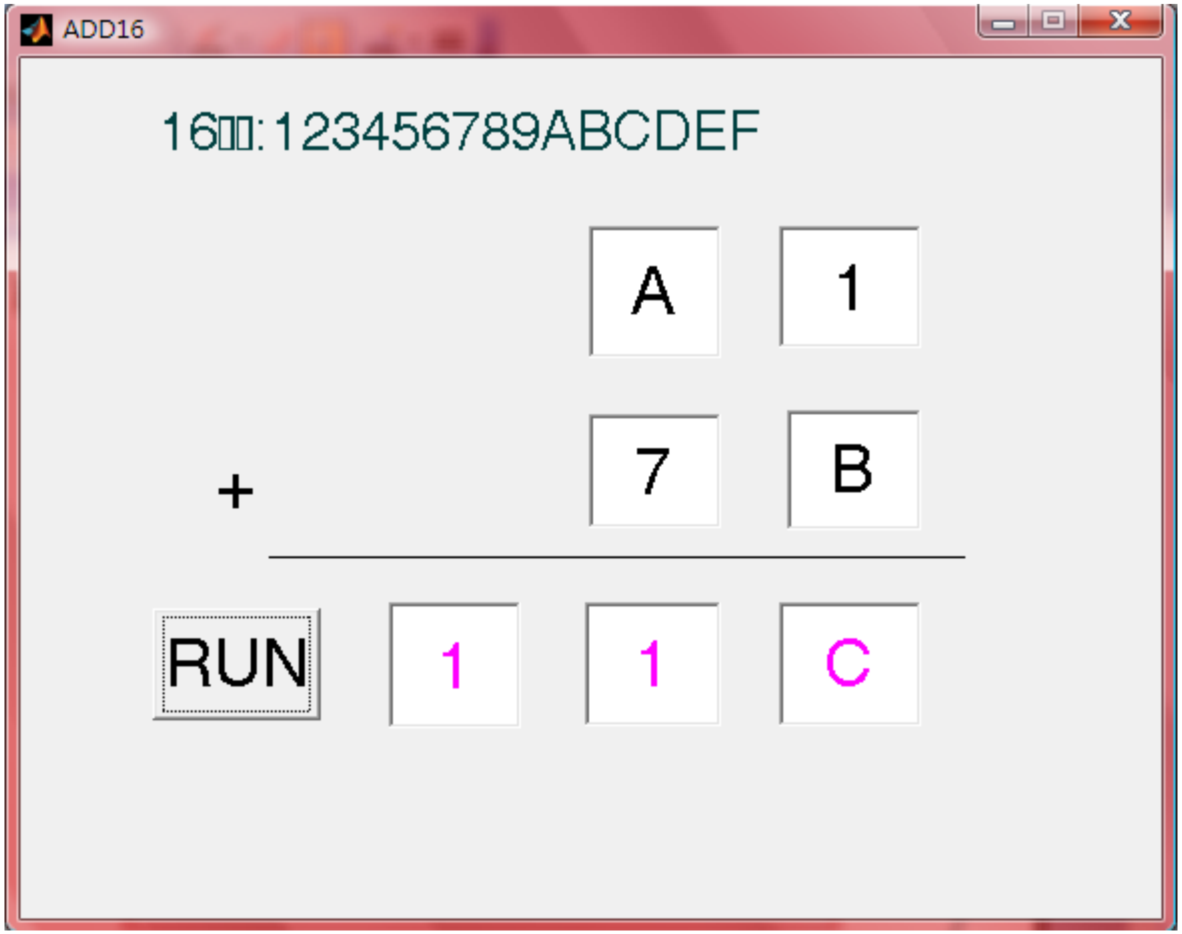
Add.fig

Add.m

ADD16

A toolbox for addition of hexadecimal numbers



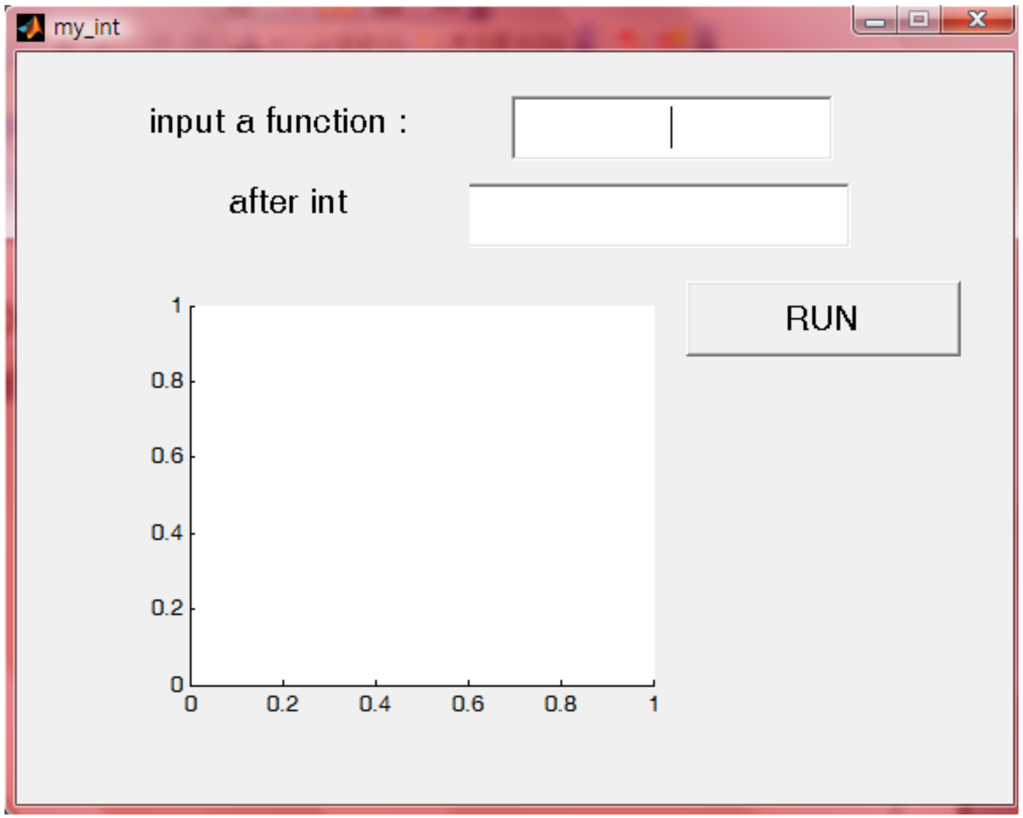

ADD16

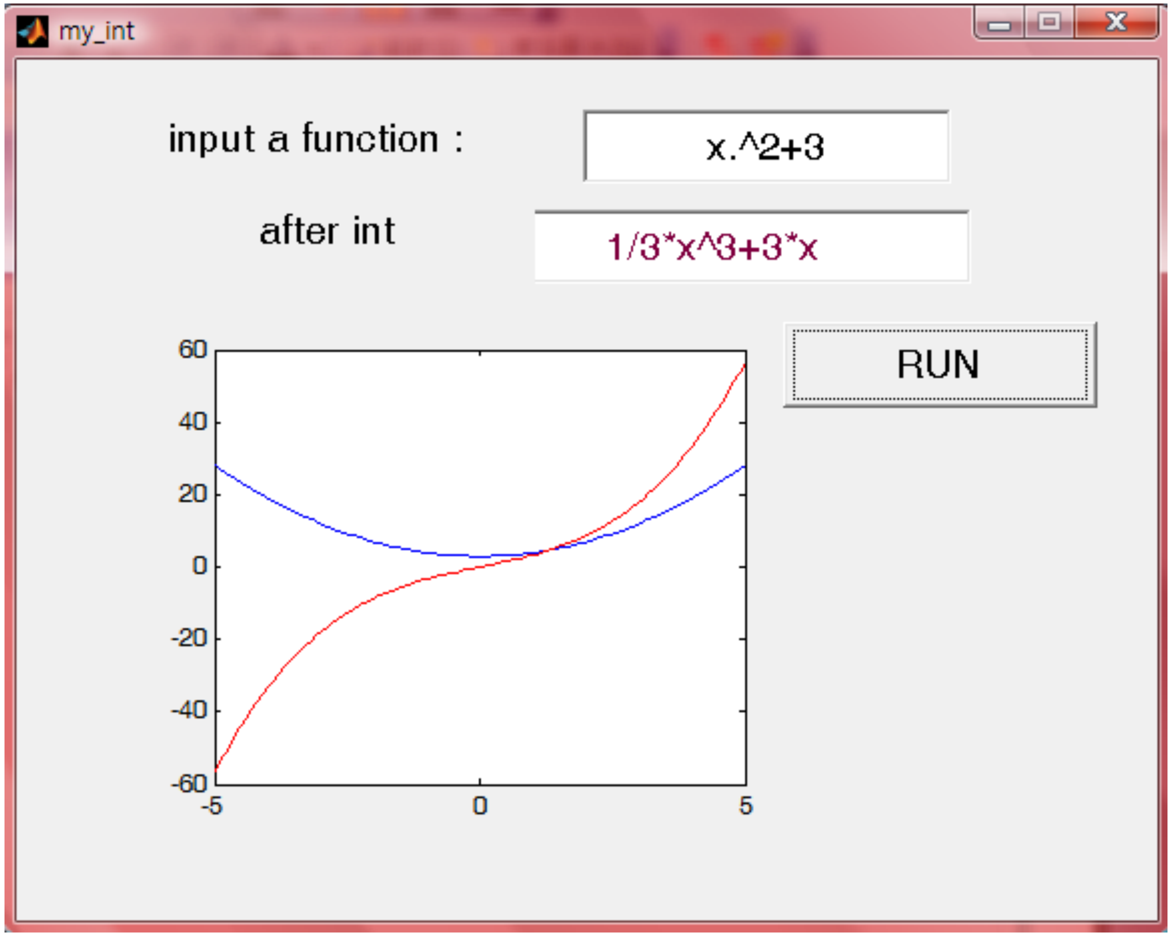
16□□:123456789ABCDEF

		A	1
+		7	B
RUN	1	1	C

Symbolic integration

- A toolbox for demonstrating symbolic integration





input a function :

x.^2+3

after int

1/3*x^3+3*x

Matlab Application Platform

- Web server
- Stand-alone execution
- Mobile connection
- Matlab → C++ or C → mobile app

Exercise #a#b

- Generate four distinct characters within {'0', '1', '2', '3', '4', '5','6','7','8','9'} randomly
- Draw a while-loop to realize the game of #a#b
  - Allow a player to key-in a four-digit string
  - Print n_a'a'n_b'b' in response to the given string
  - Halt if the guess is scored as 4'a'

- n_a denotes the number of guessed characters that appear at right position in the target
- n_b denotes the number of guessed characters that appear at wrong position in the target

Example

- Target : 6481
  - Guess : 1628 Output: 0a3b
  - Guess : 1946 Output: 0a3b
  - Guess : 6283 Output: 1a1b
  - Guess : 6481 Output: 4a0b

- Draw a flow chart to illustrate how to determine n_a for given target and guess
- Draw a flow chart to illustrate how to determine n_b for given target and guess

- Write MATLAB functions to implement flow charts for #a#b

Characters & integers

```
tt = randperm(10)-1
cc=input('keyin:', 's');
tt(1) == cc(1)
tt(1) == cc(1) - '0'
```