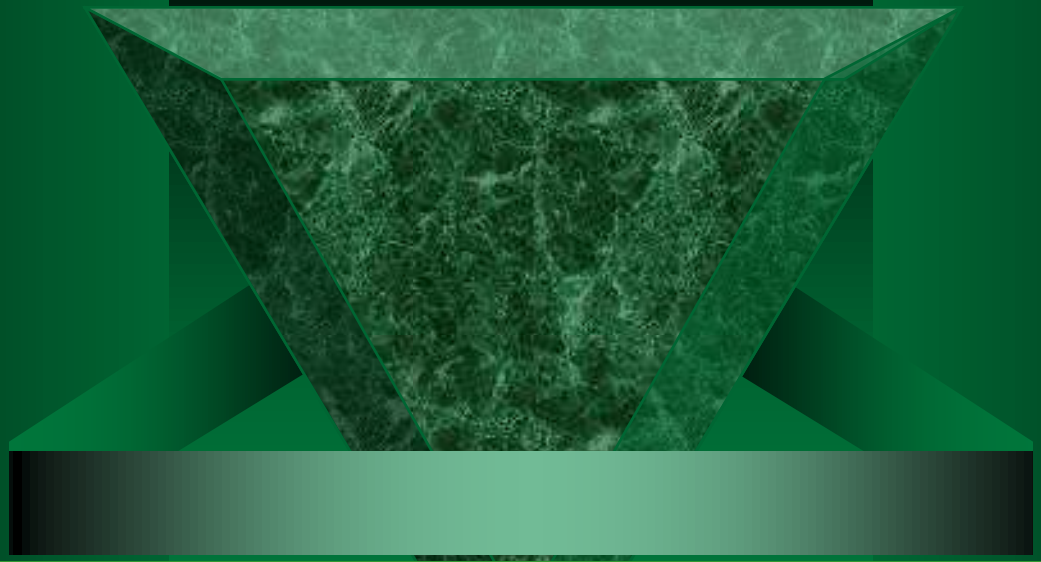
Apirak :



6.1. Subroutine

The Objectives

- Build structured programs that are divided into function
- Describe the flow of execution in a program.
- Describe what the “scope of variable” means?
- Pass data to functions
- Get value from function using return

main.pas

```pascal
program NumberSeries;
var
  choice, i: integer;

begin

  repeat
    writeln('Which series do you wish to display?');
    writeln('1 - Odd numbers from 1 to 30');
    writeln('2 - Even numbers from 1 to 30');
    writeln('3 - All numbers from 1 to 30');
    write('Enter your choice (1-3): ');
    readln(choice);

    if (choice < 1) or (choice > 3) then
      writeln('Choice must be 1, 2, or 3');

  until (choice >= 1) and (choice <= 3);

  case choice of
    1: begin
         for i := 1 to 30 do
           if (i mod 2 <> 0) then
             write(i, ' ');
         writeln;
       end;
    2: begin
         for i := 1 to 30 do
           if (i mod 2 = 0) then
             write(i, ' ');
         writeln;
       end;
    3: begin
         for i := 1 to 30 do
           write(i, ' ');
         writeln;
       end;
  end;

end.
```

ให้อ่านโปรแกรมนี้ว่าทำอะไร? แลพมีกิจกรรมอะไรบ้าง

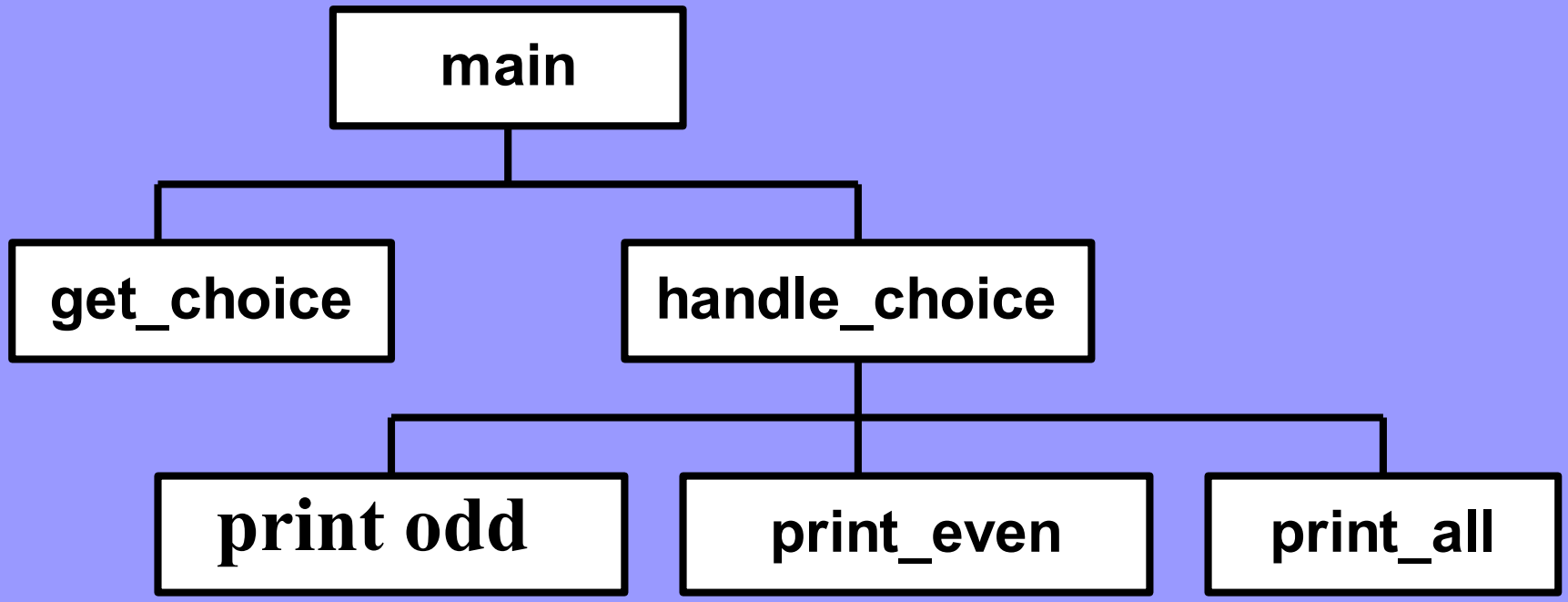
NumberSeries.pas

ตัวอย่าง 1

many activities contained in one program (complexity)

โปรแกรมรับข้อมูล 1(พิมพ์เลขคี่),2(พิมพ์เลขคู่,3(พิมพ์ทั้หมด)

```
Which series do you wish to display?
1 - Odd numbers from 1 to 30
2 - Even numbers from 1 to 30
3 - All numbers from 1 to 30
Enter your choice (1-3): 0
Choice must be 1, 2, or 3
Which series do you wish to display?
1 - Odd numbers from 1 to 30
2 - Even numbers from 1 to 30
3 - All numbers from 1 to 30
Enter your choice (1-3): 1
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29
```



- *This example if we divide our program to sub-activities like above diagram. it make easier.*
- *C++ supports the idea of dividing program to sub-activities,called function.*

ตัวอย่าง 2

Duplicate activity

ให้นักศึกษาเขียนโปรแกรมขายสินค้าและพิมพ์ใบเสร็จรับเงิน
โดยข้อมูลในระบบมีไม่เกิน **100** รายการ
ข้อมูลขายสินค้าด้วย

- รหัสสินค้า(id) เช่น “AB001”
- ชื่อสินค้า(name) เช่น “CAFÉ”
- ราคาขาย(cost) เช่น 30.6
- จำนวนสินค้าในสต๊อก (instock) เช่น 5

```
        Menu
1.      Add item to Cart
2.      Sell Product
3.      Check Stock
4.      End Program
Please select 1 / 2 / 3/ 4 =?
```

EX1 จากตัวอย่างข้างต้นให้ นักศึกษาปรับโปรแกรม โดยให้มีการตรวจดังนี้

กรณี กด 1 : ให้ตรวจสอบว่ารหัสที่ป้อน มีซ้ำหรือไม่ ถ้าไม่มีจึงจะอนุญาตให้ผู้ใช้ป้อนข้อมูล

กรณี กด 2: ให้ตรวจสอบว่ารหัสสินค้าที่ผู้ใช้ป้อนมีอยู่ในระบบหรือไม่ กรณีไม่พบให้แสดงข้อความ “product not exist”

กรณี กด 3 ให้แสดงรายการทั้งหมด และแสดงยอดรวมของ cost

Check existing products activity

แบบฝึกหัด 2 BankAccount.pas

จงเขียนโปรแกรมต้องการนำระเบียนลูกค้า(customer) ของสหกรณ์ออมทรัพย์รามคำแหง ที่รองรับการจัดเก็บข้อมูลไม่เกิน 500 คน ระเบียนลูกค้าประกอบด้วย รหัสบัญชี(id) ชื่อบัญชี(name) ที่อยู่(Address) และเงินฝาก (Deposit)

Menu

1. Open a new Account
2. Deposit
3. Withdraw
4. Compound interest
5. Quit

Please select 1 / 2 / 3 / 4 / 5 =?

เลือก 1 เป็นการเปิดบัญชีใหม่

ระบบจะแสดงหน้าจอให้ผู้ใช้ ป้อนรหัสบัญชี(id) ชื่อบัญชี(name) ที่อยู่(Address) และเงินฝาก (Deposit) โดยรหัสบัญชีต้อง ไม่ซ้ำกัน

เลือก 2 เป็นการฝากเงิน

โดยผู้ใช้ป้อนรหัสบัญชี(id)ที่ต้องการ ระบบจะตรวจสอบ โดยการค้นหาระเบียนลูกค้า เพื่อให้ผู้ใช้งานป้อนยอดฝาก เพื่อปรับปรุงบัญชีให้ถูกต้อง

เลือก 3 เป็นการถอนเงิน

โดยผู้ใช้ป้อนรหัสบัญชี(id)ที่ต้องการ ระบบจะตรวจสอบเพื่อค้นหาระเบียนลูกค้า โดยการถอนเงินลูกต้องมีเงินติดบัญชีอย่างน้อย 100 บาท ระบบทำการปรับปรุงบัญชีให้ถูกต้อง

เลือก 4 การคิดดอกเบี้ยทบต้น

1. โดยระบบจะให้ผู้ใช้ป้อนอัตราดอกเบี้ยต่อปี(rate) ทางแป้นพิมพ์ เพื่อคิดดอกเบี้ยทบต้นของทุกบัญชี
2. โปรแกรมจะแสดงเงินฝากเก่า , ดอกเบี้ยที่ได้รับ และเงินฝากใหม่

เงินฝากใหม่ = เงินฝากเดิม+ดอกเบี้ยที่ได้

ดอกเบี้ยที่ได้ =อัตราดอกเบี้ยที่ได้*เงินฝากเดิม

เลือก 5 เป็นการจบการทำงานซ้ำ

main.pas

BankAccount.pas

```
program BankAccount;

type
  Account = record
    id: string[10];
    name: string[100];
    addr: string[100];
    deposit: real;
  end;

var
  cust: array[1..500] of Account;
  n, choice, i: integer;
  id: string[10];
  bfound: boolean;
  deposit: real;

begin
  n := 0;
  repeat
    writeln;
    writeln('1. Open Acc');
    writeln('2. Deposit Acc');
    writeln('3. Withdraw Acc');
    writeln('4. Compute Interest');
    writeln('5. Exit');
    write('Enter choice: ');
    readln(choice);
```

```
    case choice of
      1:
        begin
          writeln;
          writeln('Do Open Acc');
          write('id Acc =? ');
          readln(id);
          bfound := False;

          for i := 1 to n do
          begin
            if cust[i].id = id then
            begin
              bfound := True;
              break;
            end; // end if
          end; //end for

          if  bfound = False then
            begin
                n := n + 1;
                cust[n].id := id;
                write('name Acc =? ');
                readln(cust[n].name);
                write('addr Acc =? ');
                readln(cust[n].addr);
                write('deposit Acc =? ');
                readln(cust[n].deposit);
            end
          else
            writeln('id Acc found = ', id);

        end; //case 1
```

Apirak :

BankAccount.pas

```
    2:
      begin
        writeln;
        writeln('Do Deposit Acc');
        write('id Acc =? ');
        readln(id);

        bfound := False;
        for i := 1 to n do
        begin
          if cust[i].id = id then
          begin
            bfound := True;
            break;
          end; //end if
        end; // end for

        if bfound = False then
          writeln('id Acc not found = ', id)
        else
         begin
            writeln('Before deposit Acc = ', cust[i].deposit:0:2);
            write('New deposit Acc =? ');
            readln(deposit);
            cust[i].deposit := cust[i].deposit + deposit;
            writeln('After deposit Acc = ', cust[i].deposit:0:2);
         end;
      end; //case 2

        3:
          begin
            writeln('Do Withdraw Acc');
          end;

        4:
          begin
            writeln('Do Interest Acc');
          end;
      end;

    until choice = 5;
end.
```

Apirak :

Duplicate Code

BankAccount.pas

```
37          bfound := False;
38
39          for i := 1 to n do
40          begin
41            if cust[i].id = id then
42            begin
43              bfound := True;
44              break;
45            end; // end if
46          end; //end for
47
48          if  bfound = False then
49            begin
50                n := n + 1;
51                cust[n].id := id;
52                write('name Acc =? ');
53                readln(cust[n].name);
54                write('addr Acc =? ');
55                readln(cust[n].addr);
56                write('deposit Acc =? ');
57                readln(cust[n].deposit);
58            end
```

```
71          bfound := False;
72          for i := 1 to n do
73          begin
74            if cust[i].id = id then
75            begin
76              bfound := True;
77              break;
78            end; //end if
79          end; // end for
80
81          if bfound = False then
82            writeln('id Acc not found = ', id)
83          else
84           begin
85              writeln('Before deposit Acc = ', cust[i].de
86              write('New deposit Acc =? ');
87              readln(deposit);
88              cust[i].deposit := cust[i].deposit + deposi
89              writeln('After deposit Acc = ', cust[i].dep
90           end;
```

EX1 จากตัวอย่างข้างต้นให้ นักศึกษาปรับโปรแกรม โดยให้มีการตรวจดังนี้

กรณี กด 1 : ให้ตรวจสอบว่ารหัสที่ป้อน มีซ้ำหรือไม่ ถ้าไม่มีจึงจะอนุญาตให้ผู้ใช้ป้อนข้อมูล

กรณี กด 2: ให้ตรวจสอบว่าสินค้าที่ผู้ใช้ป้อนมีอยู่ในระบบหรือไม่ กรณีไม่พบให้แสดงข้อความ “product not exist”

กรณี กด 3 ให้แสดงรายการทั้งหมด และแสดงยอดรวมของ cost

```
product List:
Id          name    cost  stock
--------------------
AB001  CAFÉ   30.6    3
```

Guidelines for building a program with function

- ***Organization**: A large program is easier to read and modify if it is logically organized into function. Once a single function is tested and performs properly,you can set it aside and concentrate on problem areas.*

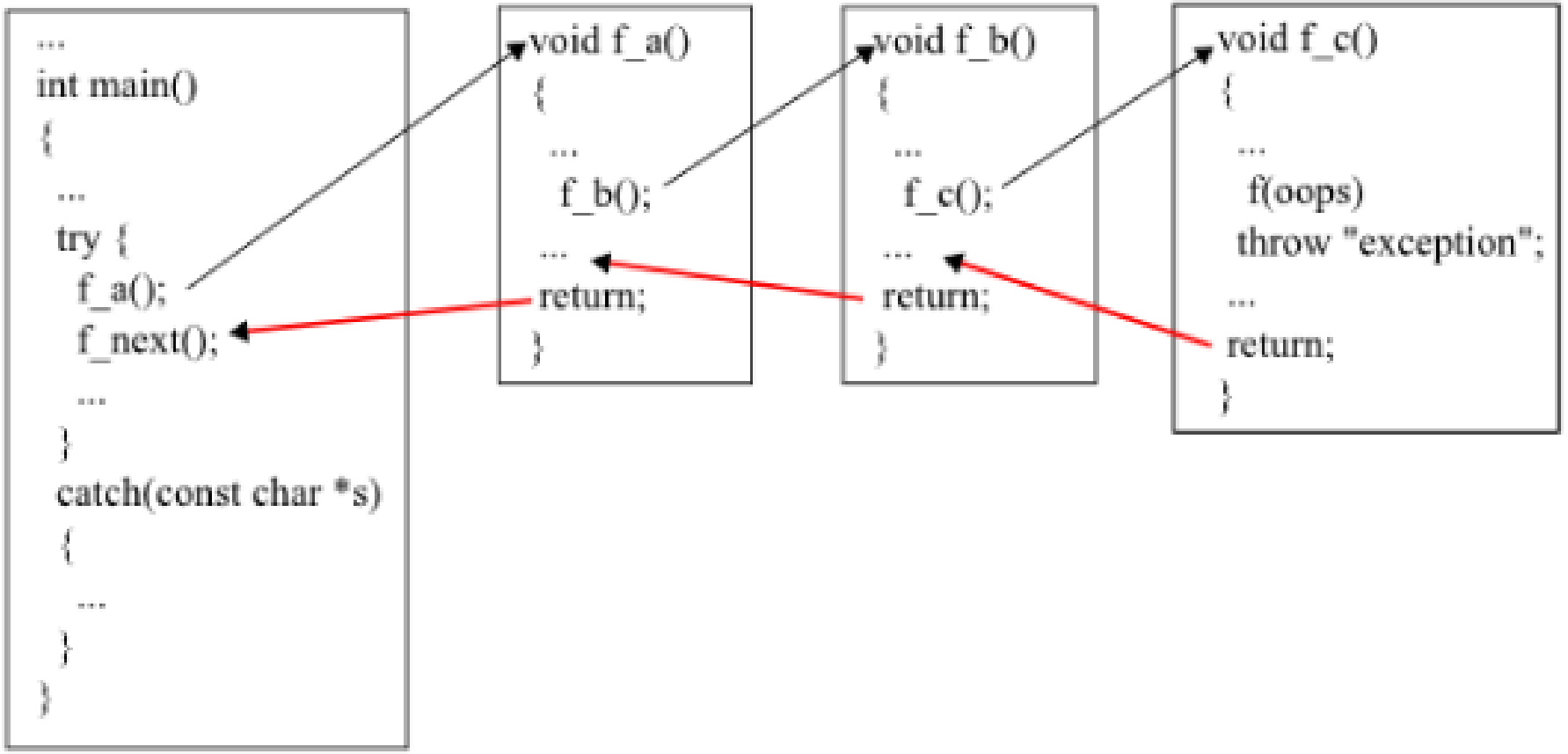
- ***Autonomy**: the function does not depend on data or code outside the function any more than necessary.*

Sub-routines in Programming lang.

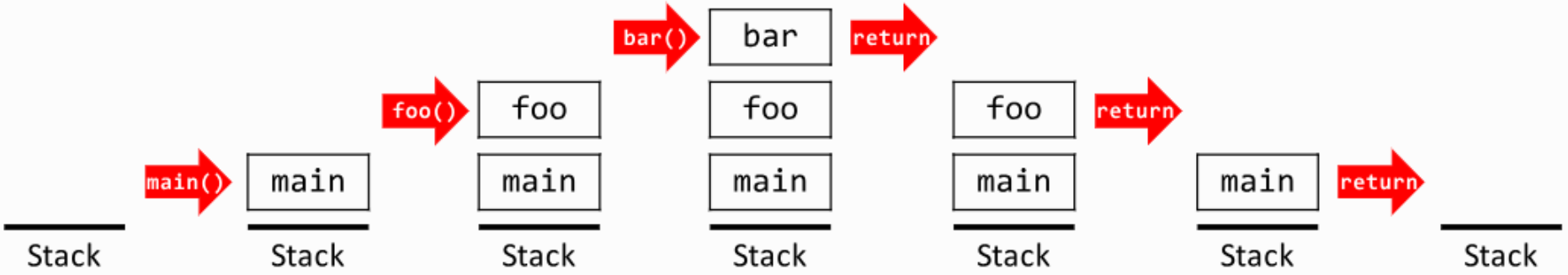
Subroutine : function or procedure

- *Library function/procedure*
  - *standard library (build-in): coming with programming lang. such as setw(-) , printf(-) in c/c++*
  - *Third-party library : coding by other programmer teams such buying , opensource.*
- *Customize function : coding by yourself such*

Execution flows (1)



Execution flows (3)



Operating System

call

main.pas

```
1  program CallExample;
2
3  procedure Bar;
4  begin
5    { empty procedure }
6  end;
7
8  procedure Foo;
9  begin
10   Bar;  { call Bar from Foo }
11 end;
12
13 begin
14   Foo;  { call Foo from main program }
15 end.
16
```

https://eecs280staff.github.io/notes/02_ProceduralAbstraction_Testing.html

Characteristic of subroutine

1. No return value subroutine (procedure)
2. Return value subroutine (function)

no return value function (procedure)

```
program <ProgramName>;

{==== Variable Declarations ====}
var
  <identifier> [, <identifier> ...] : <type>;
  { repeat as needed }

{==== Procedure Declaration ====}
procedure <ProcedureName>(<param1> : <type>; <param2> : <type>);
begin
  <statement_list>;
end;


{==== Main Program Block ====}
begin
  <statement>;
  ...
end.
```

procExa.pas

```
1  program Example;
2
3  var
4    num1:Integer;  { global }
5
6  procedure ShowResult(x: Integer);
7  begin
8    writeln('The result is: ', x);
9  end;
10
11 procedure ShowTitle();
12 begin
13   writeln('Title');
14 end;
15
16 begin
17   num1 := 10;
18   ShowTitle();
19   ShowResult(num1);
20   ShowResult(20);
21
22 end.
```

1

2

Apirak :

main.pas

```
1
2  program TennisTournament;
3
4  procedure PrintTitle(passData:string);
5  begin
6    writeln('Tennis Tournament Scheduler Program');
7    writeln('By ',passData);
8  end;
9
10 procedure PrintGoodbye;
11 begin
12   PrintTitle('PrintGoodbye');        { call to PrintGoodbye }
13   writeln('Thank you for using ');
14   writeln('the Tennis Tournament Scheduler.');
15 end;
16
17 {-----------------------------------------}
18 { Main program block }
19 {-----------------------------------------}
20 begin
21   PrintTitle('main');          { call to PrintTitle }
22   PrintGoodbye;        { call to PrintGoodbye }
23 end.
24
```

```
Tennis Tournament Scheduler Program
By main
Tennis Tournament Scheduler Program
By PrintGoodbye
Thank you for using
the Tennis Tournament Scheduler.
```

Apirak :

ProcdExit.pas

Exit -> คำสั่งที่บอกให้จบการทำงานของฟังชั้น

End fucntion

main.pas

```
1  program ProcedureWithExitExample;
2
3  procedure PrintMessage(score: Integer);
4  begin
5    if score < 0 then
6    begin
7      writeln('Invalid score!');
8      exit;   { stop procedure immediately }
9    end;
10
11   writeln('Failing grade. Try harder next time!');
12
13 end;
14
15 {-----------------------------------------}
16 { Main program block }
17 {-----------------------------------------}
18 var
19   s: Integer;
20 begin
21   PrintMessage(-10);
22   PrintMessage(10);
23   writeln('end app');
24 end.
25
```

caller

Execution flows : Function in Pascal(1)

...outine)

main.pas

```
1  program <ProgramName>;
2
3  {==== Variable Declarations ====}
4  var
5    <identifier> [, <identifier> ...] : <type>;
6
7  {==== Function Declaration ====}
8  function <FunctionName>(<param1> : <type>; <param2> : <type>) : <return_type>
9  begin
10   exit ( <expression> );
11 end;
12
13 {==== Main Program Block ====}
14 begin
15   <statement>;
16   ...
17 end.
```

nain.pas

```
1  program PowerA;
2  var
3   v1:Integer;
4  {==== Function Declaration ====}
5  function PowerA(a: Integer): Integer;
6  begin
7     { immediately return a squared value }
8    Exit(a * a);
9  end;
10
11 {==== Main Program Block ====}
12 begin
13   { call PowerA(2) → returns 4 → assign to v1 }
14   v1 := PowerA(2);
15
16 end.
17
```

2

1

3

Execution flows : Function in Pascal(2)

main.pas

PowerA.pas

```
1  program PowerA;
2
3  var
4    v1,v2: Integer;   { variable to store results }
5
6  {==== Function Declaration ====}
7  function PowerA(a: Integer): Integer;
8  begin
9    Exit(a * a);   { immediately return a squared value }
10 end;
11
12 {==== Main Program Block ====}
13 begin
14   v1 := PowerA(2);     { call PowerA(2) → returns 4 → assign to v1 }
15   WriteLn('v1 = ', v1); { output: v = 4 }
16
17   PowerA(4);     { call PowerA(4), result (16) ignored }
18
19   v2 := PowerA(6) + PowerA(8); { 36 + 64 = 100 → assign 100 to v2 }
20   WriteLn('v2 = ', v2);  { output: v2 = 100 }
21   WriteLn('powerA(20) = ', PowerA(20));  { prints "powerA(20) = 400" }
22 end.
23
```

main.pas

```
1  program ProductExample;
2  type
3    Product = record
4      id: String[10];
5      name: String[30];
6      cost: Real;
7      instock: Integer;
8    end;
9
10 var
11   newProduct: Product; {global vairable}
12
13 {==== Function Declaration ====}
14 function GetNewProduct(): Product;
15 var
16   p: Product; {local vairable}
17 begin
18   WriteLn('New product:');
19   Write('id =? ');       ReadLn(p.id);
20   Write('name =? ');     ReadLn(p.name);
21   Write('cost =? ');     ReadLn(p.cost);
22   Write('instock =? '); ReadLn(p.instock);
23   exit( p );    { return record }
24 end;
25
```

```
25
26 {==== Main Program ====}
27 begin
28  { call function returning record }
29   newProduct := GetNewProduct();
30
31   WriteLn();
32   WriteLn('--- Product Info ---');
33   WriteLn('ID: ', newProduct.id);
34   WriteLn('Name: ', newProduct.name);
35   WriteLn('Cost: ', newProduct.cost:0:2);
36   WriteLn('In stock: ', newProduct.instock);
37 end.
38
```

returnStru.pas

return structure

ข้อจำกัดในการ return array

```
1
2  program ProductExample;
3
4  var
5    G : array[1..5] of Integer;
6
7  function GetArray(): array [1...5] of Integer;
8  var
9    P : Array [1...5] of Integer; {local vairable}
10 begin
11   p[1] := 10;
12   exit( p );   { return array }
13 end;
14
15 begin
16   WriteLn;
17   G := GetArray();
18   WriteLn('g[1] ', g[1]);
19 end.
20
```

```
Free Pascal Compiler version 3.0.4 [2017/10/03] for x86_64
Copyright (c) 1993-2017 by Florian Klaempfl and others
Target OS: Linux for x86-64
Compiling main.pas
main.pas(7,22) Error: Type identifier expected
main.pas(7,22) Fatal: Syntax error, ";" expected but "ARRAY" found
Fatal: Compilation aborted
Error: /usr/local/fpc-3.0.4/bin/ppcx64 returned an error exitcode
```

HelloWorld.pas

RetArry.pas

442x6

```
1  program ReturnArray;
2
3  type
4    myArray = array[1..5] of Integer;
5
6  var
7    G1: array[1..5] of Integer;
8    G2: myArray;
9
10 function GetArray(): myArray;
11 var
12   P1: array[1..5] of Integer;  { local variable, same type shape as myArray }
13   P2: myArray;                 { local variable of declared type }
14 begin
15   { initialize array }
16   P1[1]  := 10;
17   P2[1] := 20;
18   //exit( P1 );  { return array }
19    exit( P2 );  { return array }
20 end;
21
22 begin
23   G1 := GetArray();  { call the function }
24   WriteLn('G1[1] = ', G1[1]);
25
26   G2 := GetArray();  { call the function }
27   WriteLn('G2[1] = ', G2[1]);
28
29 end.
30
```

กำหนดประเภท array เพื่อ return

HelloWorld.pas

retArayGuid.pas

แนวปฏิบัติ

1.Create array type
2.Return array type

ReturnArray

```
1  program ReturnArray;
2
3  type
4    myArray = array[1..5] of Integer;
5
6  var
7    G2: myArray;
8
9  function GetArray(): myArray;
10 var
11   P2: myArray;                    { local variable of
12 begin
13   { initialize array }
14   P2[1] := 20;
15    exit( P2 );  { return array }
16 end;
17
18 begin
19
20   G2 := GetArray();  { call the function }
21   WriteLn('G2[1] = ', G2[1]);
22
23 end.
24
```

Python allows a function return multiple values

main.py

```python
def operate(a, b):
    sum = a + b
    diff = a - b
    mul = a * b
    div = a / b
    return sum, diff, mul, div

n1 = 5
n2 = 10
sum, diff, mul, div = operate(n1, n2)
print(
    f"The sum is {sum}\n"
    f"The difference is {diff}\n"
    f"The multiplication gives {mul}\n"
    f"The division gives {div}\n"
)
```

Shell

```
The sum is 15
The difference is -5
The multiplication gives 50
The division gives 0.5
```

https://www.programiz.com/python-programming/online-compiler/

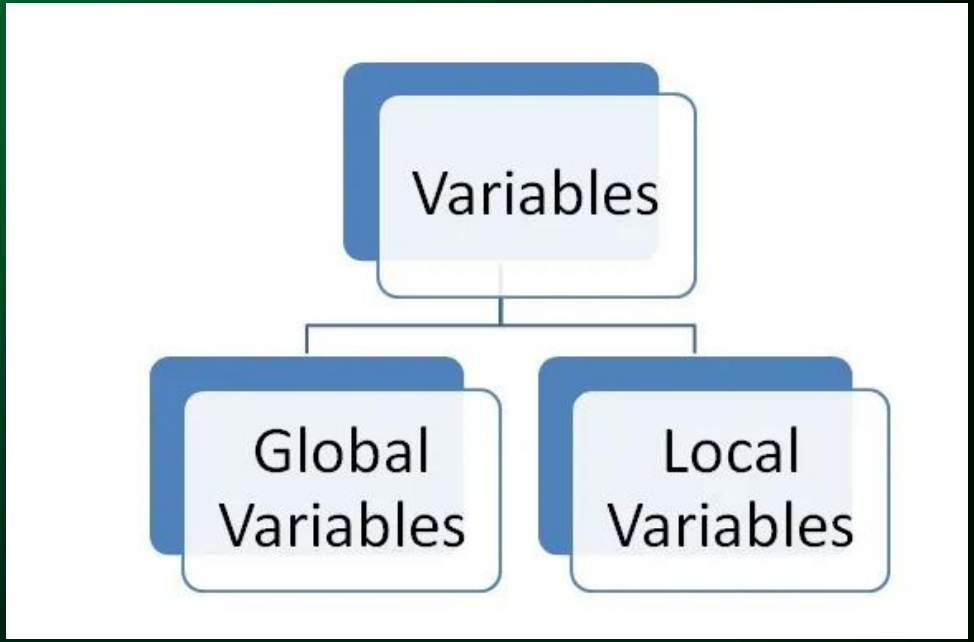
Scope of variable (Global & Local variable)

Scope determines the accessibility (visibility) of variables. c/c++ supports the following scope :

Local Block scope

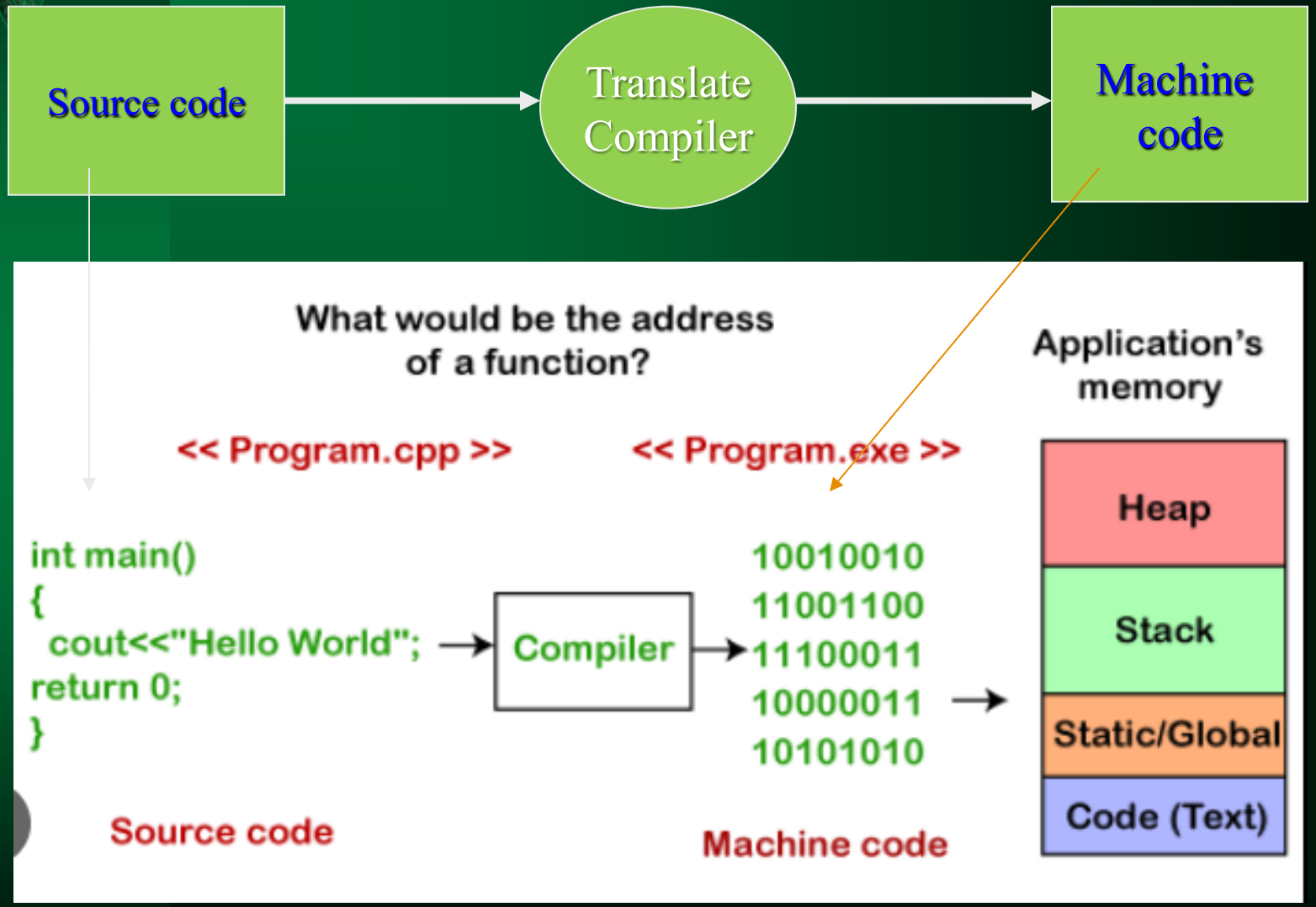
Local Function scope

Global Scope



Recap:

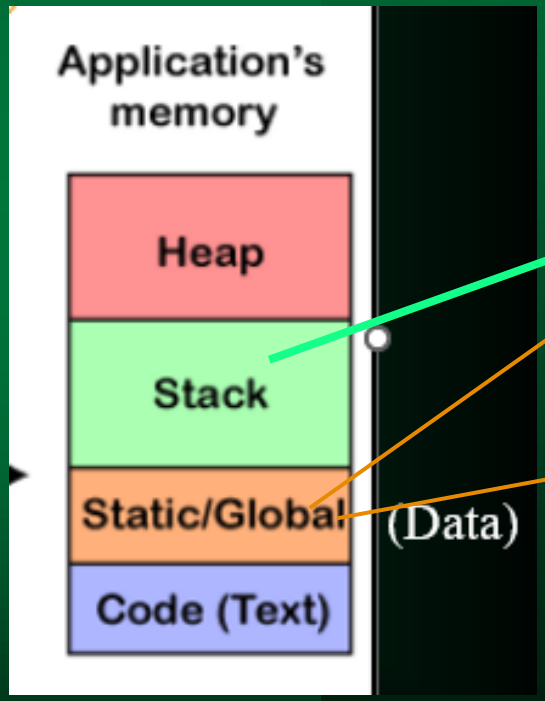
Compiler is a tool that translate a source code to machine code



(Data)

Global & Local Block & Local scope Property

- ตัวแปรlocal&paralter จะถูก allocate ใน **stack** เมื่อฟังก์ชันเริ่มทำงาน และจะ free เมื่อจบฟังก์ชัน
- ตัวแปร **global** จัดเก็บใน **data segment** ตั้งแต่เริ่มโปรแกรมจนกระทั้งจบ



main.pas

```
1  program VariableScopeExample;
2  var
3    x, y : Integer;
4
5  function AddAndCount(a, b: Integer): Integer;
6  var
7    sum: Integer;
8  begin
9    sum := a + b;
10   exit(sum);
11 end;
12
13 var
14   result: Integer;
15
16 begin
17   Write('Enter first number: ');
18   ReadLn(x);
19   Write('Enter second number: ');
20   ReadLn(y);
21   result := AddAndCount(x, y);     { call func
22
23   WriteLn('Sum = ', result);
24 end.
```

Block scope &local & parameter scope

main.pas

```
1  program VariableScopeExample;
2
3  var
4    g: Integer = 0;    // global variable
5
6  procedure f1;
7  var
8    f1_num: Integer;  // local variable
9  begin
10   g := g + 1;
11   writeln('1.1 f1_g = ', g , ' ,addr = ', PtrUInt(@g));
12   writeln('1.2 f1_num = ', f1_num, ' ,addr = ', PtrUInt(@f1_num));
13   f1_num := f1_num + 1;
14 end;
15
16 procedure f2( g:Integer ; x:Integer); //parameter
17 begin
18   writeln('2.1 f2_g = ', g, ' ,addr = ', PtrUInt(@g));
19   writeln('2.2 f2_x = ', x, ' ,addr = ', PtrUInt(@x));
20 end;
21
22 procedure f3 ();
23 var
24   g: Integer;  // local variable (ซ่อน global variable)
25 begin
26   g := 100;
27   writeln('3. f3_g = ', g, ' ,addr = ', PtrUInt(@g));
28   {f1_num := 200;}
29   {x := 200;}
30 end;
```

```
33 begin
34   {f1_num := 100;}
35   writeln('a. main_g is ', g, ' ,addr = ', PtrUInt(@g));
36   f1;
37   writeln('b. main_g is ', g);
38   f2(20,10);
39   writeln('c. main_g is ', g);
40   f3;
41   writeln('d. main_g is ', g);
42   f1;
43   writeln('e. main_g is ', g);
44 end.
45
```

vscope.pas

```
a. main_g is 0 ,addr = 4346128
1.1 f1_g = 1 ,addr = 4346128
1.2 f1_num = 0 ,addr = 140736365335580
b. main_g is 1
2.1 f2_g = 20 ,addr = 140736365335576
2.2 f2_x = 10 ,addr = 140736365335568
c. main_g is 1
3. f3_g = 100 ,addr = 140736365335580
d. main_g is 1
1.1 f1_g = 2 ,addr = 4346128
1.2 f1_num = 0 ,addr = 140736365335580
e. main_g is 2
```

main.pas

```
1  program VariableScopeExample;
2
3  var
4    g: Integer = 0;    // global variable
5
6  procedure f1;
7  var
8    f1_num: Integer;  // local variable
9  begin
10   g := g + 1;
11   writeln('1.1 f1_g = ', g , ' ,addr = ', PtrUInt(@g));
12   writeln('1.2 f1_num = ', f1_num, ' ,addr = ', PtrUInt(@f1_num));
13   f1_num := f1_num + 1;
14 end;
15
16 procedure f2( g:Integer ; x:Integer); //parameter
17 begin
18   writeln('2.1 f2_g = ', g, ' ,addr = ', PtrUInt(@g));
19   writeln('2.2 f2_x = ', x, ' ,addr = ', PtrUInt(@x));
20 end;
21
22 procedure f3 ();
23 var
24   g: Integer;  // local variable (ซ่อน global variable)
25 begin
26   g := 100;
27   writeln('3. f3_g = ', g, ' ,addr = ', PtrUInt(@g));
28   {f1_num := 200;}
29   {x := 200;}
30 end;
```

vscope.pas

```
33 begin
34   {f1_num := 100;}
35   writeln('a. main_g is ', g, ' ,addr = ', PtrUInt(@g));
36   f1;
37   writeln('b. main_g is ', g);
38   f2(20,10);
39   writeln('c. main_g is ', g);
40   f3;
41   writeln('d. main_g is ', g);
42   f1;
43   writeln('e. main_g is ', g);
44 end.
45
```

vscope.pas

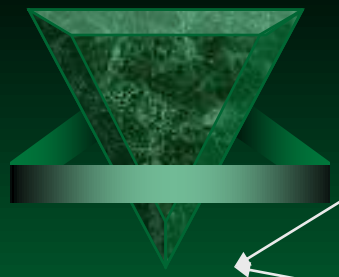
```
a. main_g is 0 ,addr = 4346128
1.1 f1_g = 1 ,addr = 4346128
1.2 f1_num = 0 ,addr = 140736365335580
b. main_g is 1
2.1 f2_g = 20 ,addr = 140736365335576
2.2 f2_x = 10 ,addr = 140736365335568
c. main_g is 1
3. f3_g = 100 ,addr = 140736365335580
d. main_g is 1
1.1 f1_g = 2 ,addr = 4346128
1.2 f1_num = 0 ,addr = 140736365335580
e. main_g is 2
```

```pascal
program GlobalLocalExample;
var
  g, i: Integer;    // global vari

procedure myfunction;
var
  k: Integer;       // local varia
  g: Integer;       // local varia
begin
  g := 40;
  k := i;
  i := i + 1;       // equivalent
  g := k;
  writeln('myfunction k :', k);
  writeln('myfunction g :', g);
end;

var
  j, k: Integer;
```

```pascal
begin
  g := 10;
  i := 3;
  j := 2;
  k := i + j;
  writeln('1.main k :', k);
  writeln('1.main i :', i);
  myfunction;
  writeln('2.main k :', k);
  writeln('2.main j :', j);
  writeln('2.main i :', i);
  g := g + 1;
  myfunction();
  writeln('3.main k :', k);
  writeln('3.main j :', j);
  writeln('3.main g :', g);
  writeln('4.main i :', i);
  readln;
end.
```

gscope.pas

k = ? j = ? g = ? i = ?



duplicate

main.pas

```
1  program GlobalLocalExample2;
2
3  var
4    g, i: Integer;  // global variables
5
6  procedure myfunction;
7  var
8    l: Integer;     // local variable
9    g: Integer;     // local variable shadows global 'g'
10 begin
11   g := 40;
12   i := i + 1;     // equivalent to ++i
13   l := i;         // l = ++i
14   writeln('l = ', l);
15   writeln('g (myfunction) = ', g);
16   g := 80;        // local 'g' only affects this procedure
17   {writeln('k = ',k); }
18 end;
19
20 var
21   j, k: Integer;  // local to main program
22
23 begin
24   g := 10;
25   i := 3;
26   j := 2;
27   k := i + j;
28   writeln('before g = ', g);
29   writeln('j = ', j, ' and k = ', k);
30   writeln('i = ', i, ' before the call to myfunction.');
31   myfunction;
32   writeln('i = ', i, ' after the call to myfunction.');
33   writeln('after g = ', g);
34   readln;
35 end.
```

gscope2.pas

ผลของการ execute โปรแกรม คือ ?

7.2 parameter passing (1)

Passing/Getting Data to and from function

- There are two ways to pass data to function in Pascal
  - Passing by value : send data to process on a function
  - Passing by reference : retrieve/update data form a function

7.2 parameter passing (2)

วัตถุประสงค์ของ PassbyValue & PassbyReference

- PassbyValue : วัตถุประสงค์เพื่อต้องการส่งข้อมูลไปยัง subrountine เพื่อนำไปใช้งาน เช่น การแสดงผล , การคำนวณต่างๆ

- PassbyReferecne : วัตถุประสงค์เพื่อต้องการ update ค่าตัวแปร (variable) ผ่าน subroutine

Grammar : Function Body

passValRef.pas

```
1
2  program PassByValueAndReference;
3
4  procedure pass_val( l: Integer;  var y: real );
5  begin
6    writeln(chr(9),'1.pass_val->l = ', l:0);
7    writeln(chr(9),'1.pass_val->y = ', y:0:1);
8    l := l * 2;
9    y := y + 2;
10   writeln(chr(9),'2.pass_val->l = ', l:0);
11   writeln(chr(9),'2.pass_val->y = ', y:0:1);
12 end;
13
14 var
15   x: Integer;
16   y: real;
17
18 begin
19   y := 18.2;
20   x := 3;
21   writeln('before->x = ', x);
22   writeln('before->y = ', y:0:1);
23
24   pass_val( x, y );
25
26   writeln('before->x = ', x);
27   writeln('before->y = ', y:0:1);
28   readln;
29 end.
```

parameters

-x = passed by values
-y = pass by reference

arguments

เปรียบระหว่าง PrintData1&PrintData2

main.pas

```
1  program whichOne;
2
3  procedure PrintData1( x: integer);
4  begin
5    writeln( ' x = ' , x);
6  end;
7
8  procedure PrintData2(var x: integer);
9  begin
10   writeln( ' x = ' , x);
11 end;
12
13 var
14  data:integer;
15
16 begin
17   write( 'data (integer) = ' );
18   read( data);
19   PrintData1( data );
20   PrintData2( data );
21
22 end.
23
```

whichOne.pas

7.2 Passing Data

- Memory of parameters on function are allocated on stack.
- They are allocated during a function is executing ,and are deallocated after the function is ended like local parameters.
- The initial values of parameters is from argument variables.

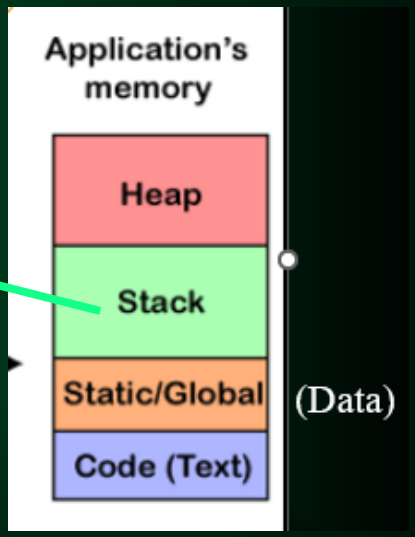
Grammar : Function Body

```
1
2  program PassByValueAndReference;
3
4  procedure pass_val( l: Integer;  var y: real );
5  begin
6    l := l * 2;
7    y := y + 2;
8    writeln(chr(9),'pass_val->l = ', l:0);
9    writeln(chr(9),'pass_val->y = ', y:0:1);
10 end;
11
12 var
13   x: Integer;
14   y: real;
15
16 begin
17   y := 18.2;
18   x := 3;
19   writeln('before->x = ', x);
20   writeln('before->y = ', y:0:1);
21
22   pass_val( x, y );
23
24   writeln('before->x = ', x);
25   writeln('before->y = ', y:0:1);
26   readln;
27 end.
```

parameters

-x = passed by values
-y = pass by reference

arguments

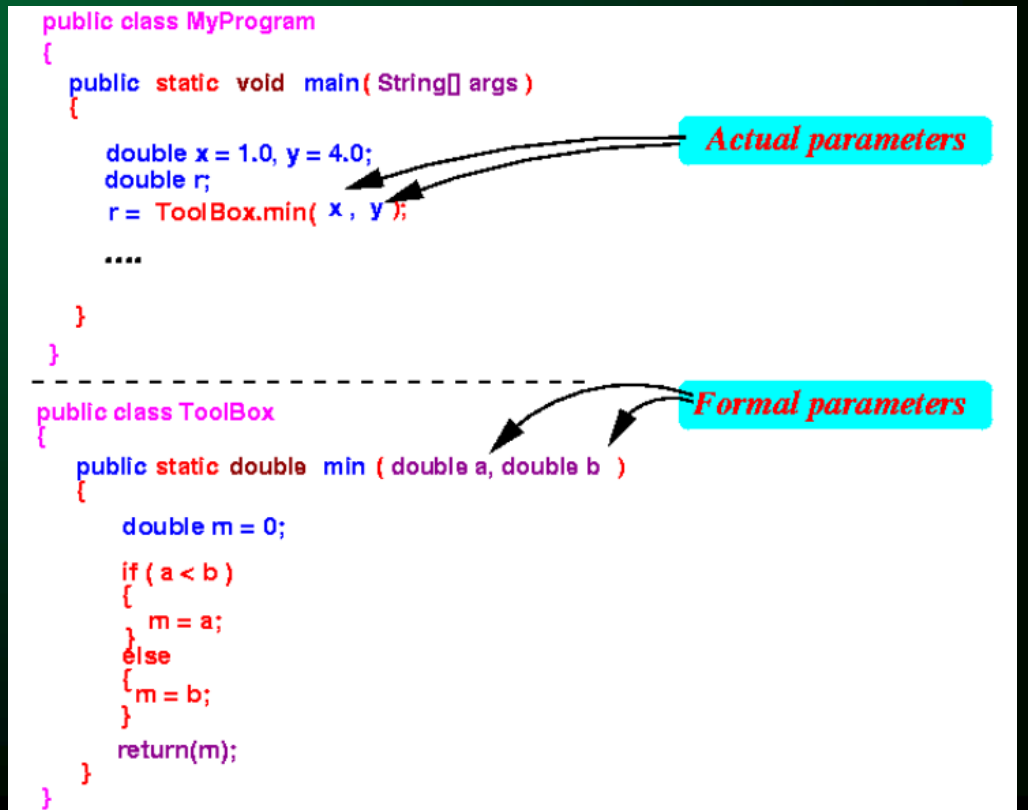


7.2 Passing Data *Pass by value*

- The objective of design function with “pass by value” is the sending data from the caller to process in that function routine.

- The mechanism is the copying values of argument variables to be the initial value of parameter variables, so updating parameter variables in the calling function the argument value remain the same.

Parameter : formal parameters
Argument : actual parameters

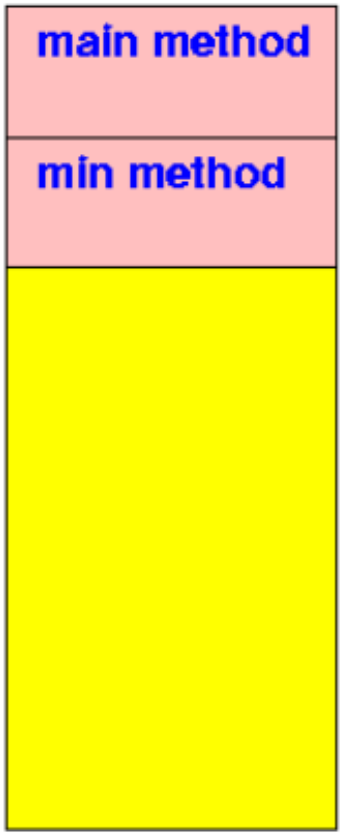


https://www.cs.emory.edu/~cheung/Courses/170/Syllabus/08/parameter2.html

ตย1. กลไกล pass by value

```
public class MyProgram
{
   public  static  void  main ( String[] args )
   {
      double x = 1.0, y = 4.0;
      double r;
      r =  ToolBox.min( x , y );
      ....
   }
}
```

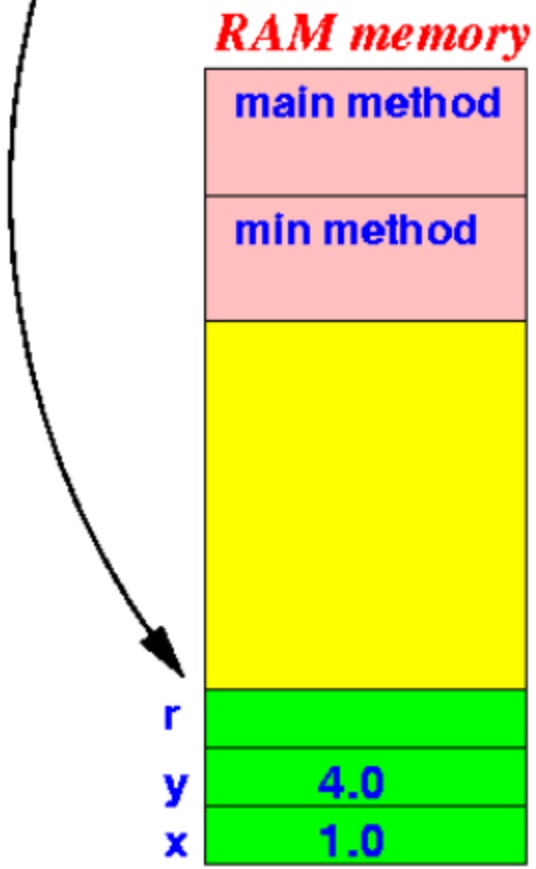
```
public class ToolBox
{
    public static double  min ( double a, double b  )
    {
        double m = 0;
        if ( a < b )
        {
           m = a;
        }
        else
        {
           m = b;
        }
        return(m);
    }
}
```



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

      double x = 1.0, y = 4.0;
      double r;
      r =  ToolBox.min( x , y );

      ....

   }
}
```

```
public class ToolBox
{
    public static double  min ( double a, double b  )
    {
        double m = 0;

        if ( a < b )
        {
           m = a;
        }
        else
        {
           m = b;
        }
        return(m);
    }
}
```

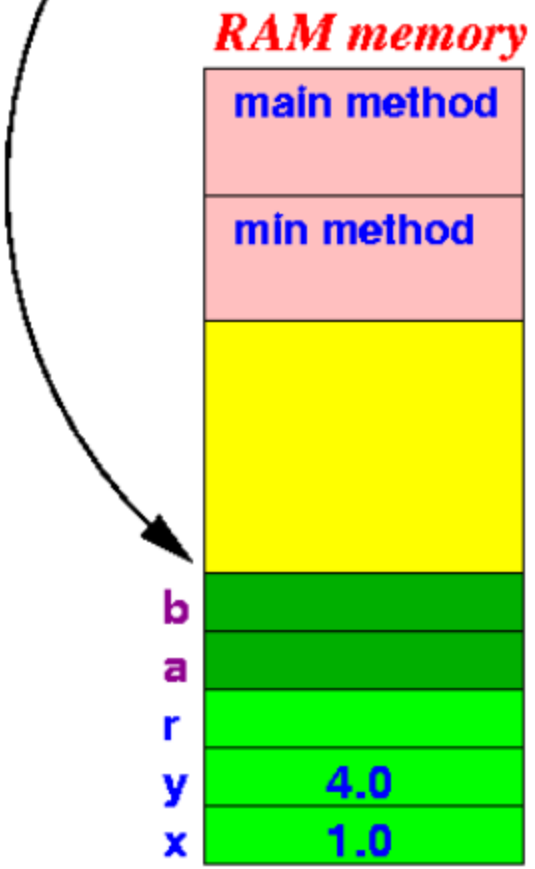
main creates its local variables



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

      double x = 1.0, y = 4.0;
      double r;
      r =  ToolBox.min(  x ,  y );

      ....

   }
}
```

```
public class ToolBox
{
    public static double  min ( double a, double b  )
    {
        double m = 0;

        if ( a < b )
        {
           m = a;
        }
        else
        {
           m = b;
        }
        return(m);
    }
}
```

Pass-by-value:
1. create parameter variables

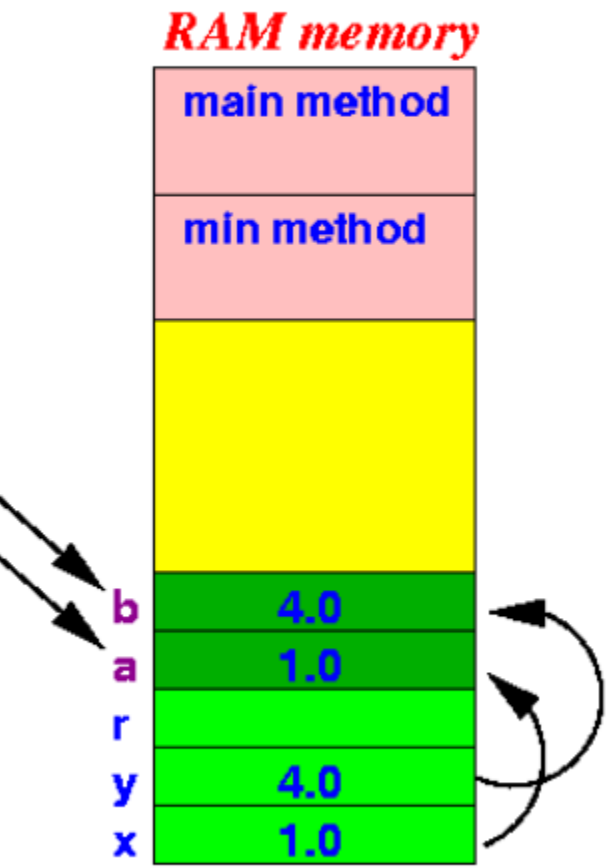


```
public class MyProgram
{
   public  static  void  main( String[] args )
   {
        double x = 1.0, y = 4.0;
        double r;
        r =  ToolBox.min( x , y );
        ....
   }
}
```

```
public class ToolBox
{
    public static double  min ( double a, double b )
    {
         double m = 0;
         if ( a < b )
         {
            m = a;
         }
         else
         {
            m = b;
         }
         return(m);
    }
}
```

Pass-by-value:

2. copy value of actual parameter to formal parameter



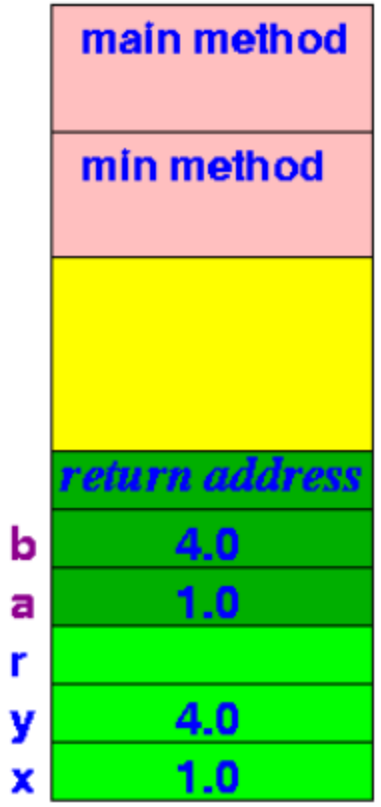
Method invocation mechanism: Jump to method

```
public class MyProgram
{
   public  static  void  main( String[] args )
   {
       double x = 1.0, y = 4.0;
       double r;
       r =  ToolBox.min( x , y );
       ....
   }
}
```

return address

```
public class ToolBox
{
    public static double  min ( double a, double b  )
    {
        double m = 0;
        if ( a < b )
        {
          m = a;
        }
        else
        {
          m = b;
        }
        return(m);
    }
}
```

RAM memory



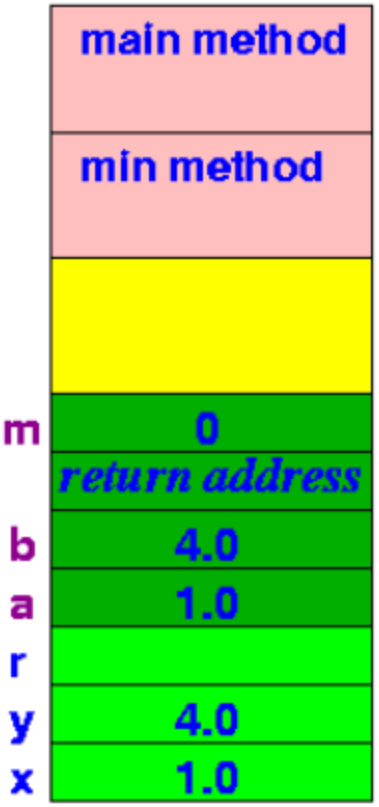
```
public class MyProgram
{
   public  static  void  main( String[] args )
   {

      double x = 1.0, y = 4.0;
      double r;
      r =  ToolBox.min( x , y );

      ....

   }
}
```

Method invocation mechanism: Jump to method

return address

```
public class ToolBox
{
    public static double  min ( double a, double b  )
    {
        double m = 0;

        if ( a < b )
        {
          m = a;
        }
        else
        {
          m = b;
        }
        return(m);
    }
}
```

RAM memory



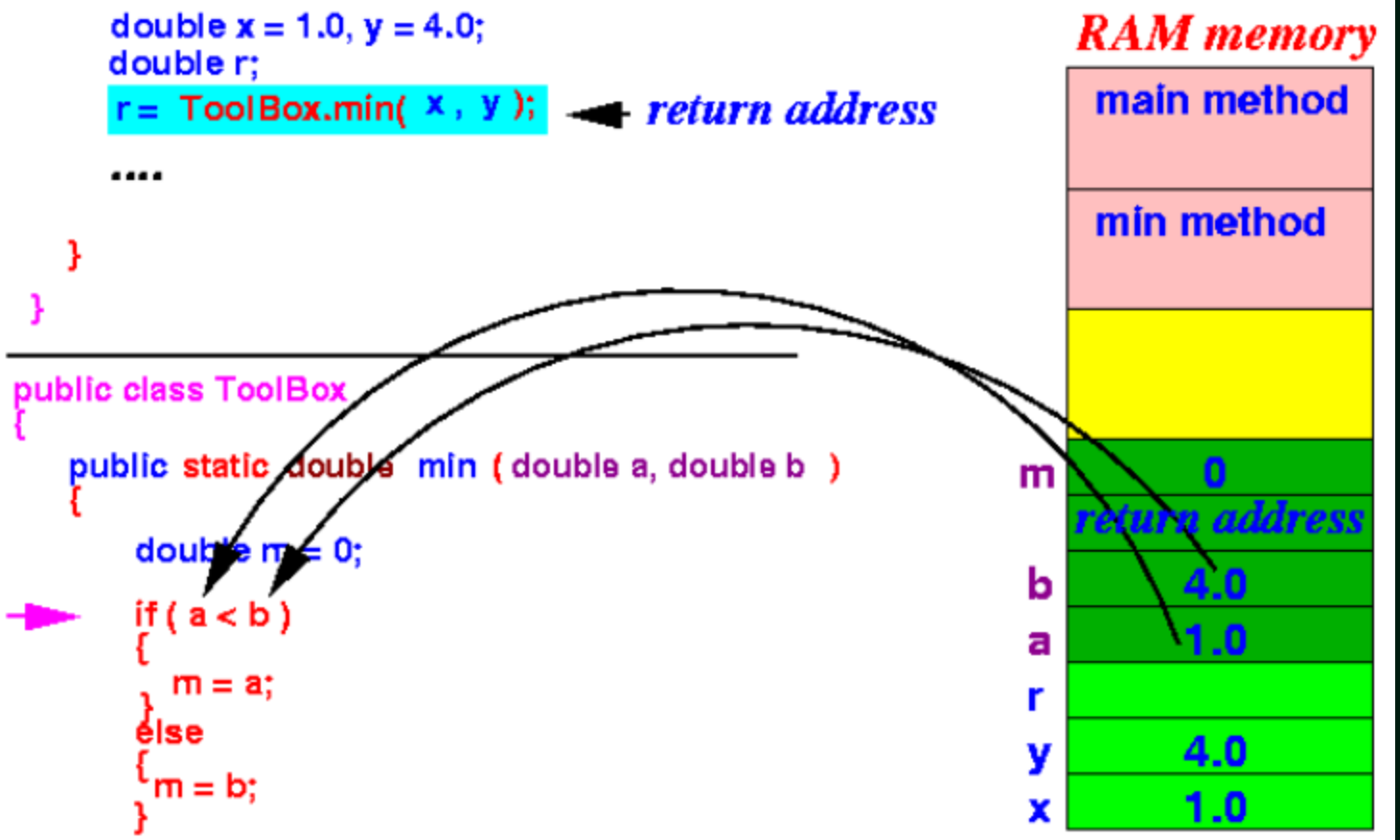
create local variable

```
public class MyProgram
{
   public  static  void  main ( String[] args )
   {
       double x = 1.0, y = 4.0;
       double r;
       r =  ToolBox.min( x , y );
       ....

   }
}
```

Method access information by:
Obtain information from parameter variable

return address

```
public class ToolBox
{
    public  static  double  min  ( double a, double b  )
    {
        double m = 0;
        if ( a < b )
        {
           m = a;
        }
        else
        {
           m = b;
        }
        return(m);
    }
}
```



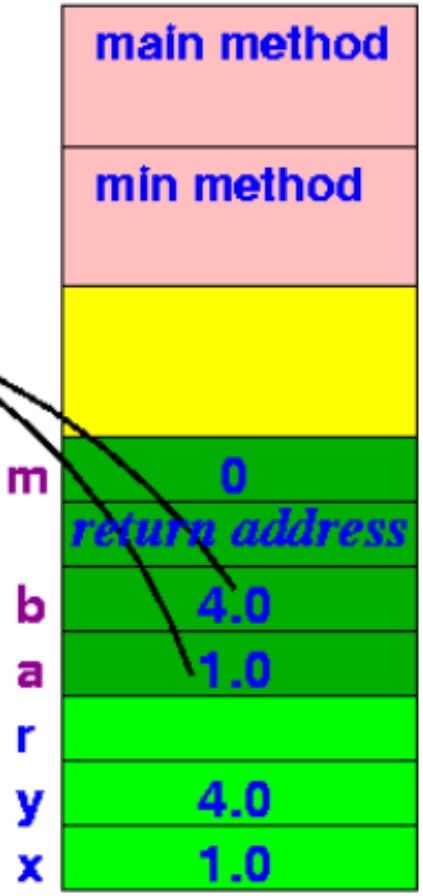
```
public class MyProgram
{
   public  static  void  main ( String[] args )
   {
       double x = 1.0, y = 4.0;
       double r;
       r =  ToolBox.min( x , y );
       ....
   }
}
```

return address

Method access information by: Obtain information from parameter variable

```
public class ToolBox
{
    public static double  min ( double a, double b  )
    {
        double m = 0;
        if ( a < b )
        {
           m = a;
        }
        else
        {
           m = b;
        }
        return(m);
    }
}
```

RAM memory



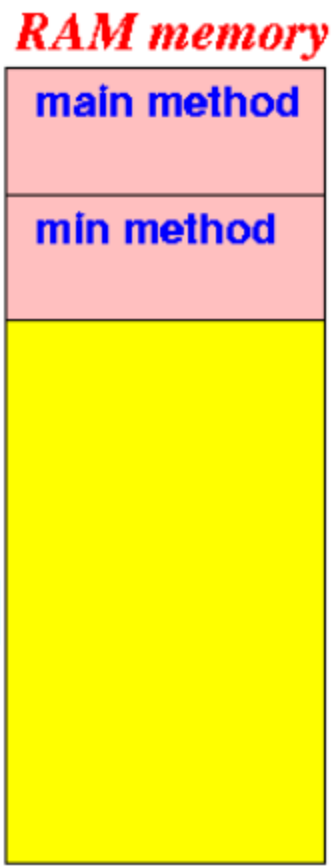
Ex2 :

```
public class MyProgram
{
   public  static  void  main ( String[] args )
   {
       double x = 1.0, y = 4.0;
       double r;
       r =  ToolBox.fun(  x ,  y );
       System.out.println(x);
       System.out.println(y);
       System.out.println(r);
   }
}
```

?

```
public class ToolBox
{
    public static double  fun  ( double a, double b  )
    {
        double m = 0;
        a = a + 1;
        b = b + 2;
        m = a + b;
        return(m);
    }
}
```

Update parameter variable !

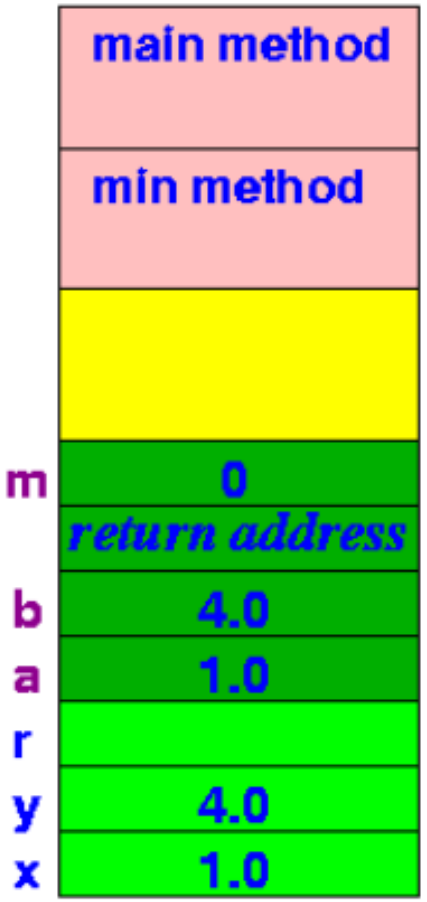


```
public class MyProgram
{
   public  static  void  main( String[] args )
   {
        double x = 1.0, y = 4.0;
        double r;
        r =  ToolBox.fun(  x ,  y );
        System.out.println(x);
        System.out.println(y);
        System.out.println(r);
   }
}
```

```
public class ToolBox
{
    public static double  fun  ( double a, double b  )
    {
        double m = 0;
        a = a + 1;
        b = b + 2;
        m = a + b;
        return(m);
    }
}
```

Situation when method fun starts running

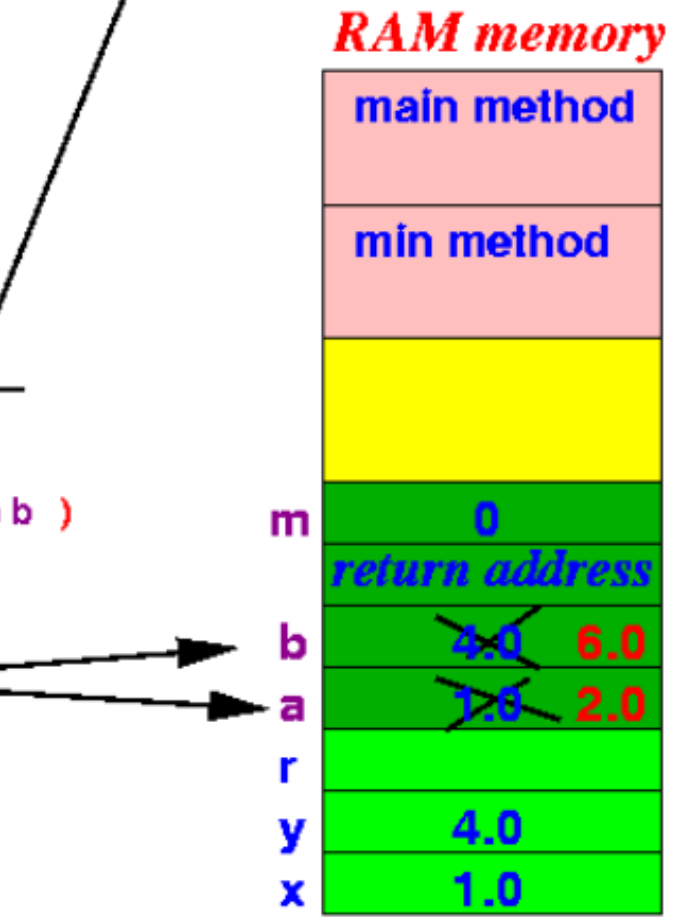
RAM memory



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

       double x = 1.0, y = 4.0;
       double r;
       r =  ToolBox.fun(  x ,  y );

       System.out.println(x);
       System.out.println(y);
       System.out.println(r);
   }
}
```

```
public class ToolBox
{
    public static double  fun  ( double a, double b  )
    {
        double m = 0;
        a = a + 1;
        b = b + 2;
        m = a + b;
        return(m);

    }
}
```

Assignment statements changes values stored in parameter var



- The *values* in the *actual parameters* (x and y) are unchanged !!!

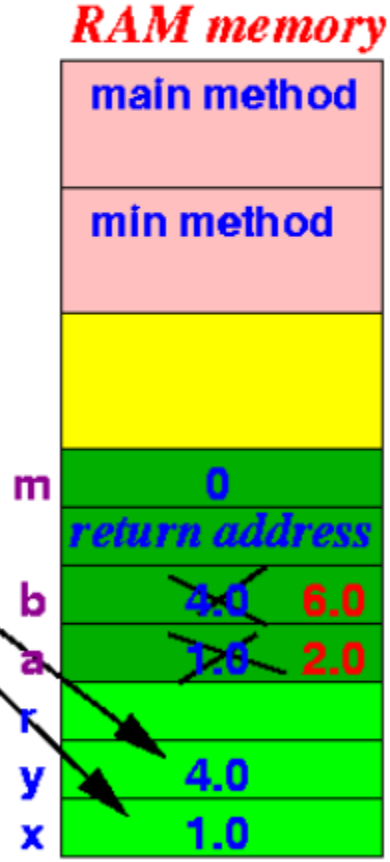
```
public class MyProgram
{
   public  static  void  main( String[] args )
   {
       double x = 1.0, y = 4.0;
       double r;
       r =  ToolBox.fun(  x ,  y );
       System.out.println(x);
       System.out.println(y);
       System.out.println(r);
   }
}
```

```
public class ToolBox
{
    public static double  fun  ( double a, double b )
    {
        double m = 0;
        a = a + 1;
        b = b + 2;
        m = a + b;
        return(m);
    }
```

That's why the **statements**

```
System.out.println(x);    ---> prints 1.0
System.out.println(y);    ---> prints 4.0
```

Actual parameters not affected !!



Passing by value in Pascal:

PassVal.pas

```
1  program PassByValueDemo;
2
3  procedure PassVal(a: integer; b: real);
4  begin
5      a := a * 2;
6      b := b + 2;
7      writeln('a = ', a , ', address of a = ' , HexStr(@a));
8      writeln('b = ', b:0:1 , ', address of b = ' , HexStr( @b ));
9  end;
10
11 var
12     x: integer;
13     y: real;
14
15 begin
16     x := 3;
17     y := 18.0;
18     PassVal(x, y);
19     writeln('x = ', x , ', address of x = ' , HexStr(@x));
20     writeln('y = ', y:0:1 , ', address of y = ' , HexStr( @y ));
21 end
```

```
a = 6, address of a = 00007FFE3FEDAF68
b = 20.0, address of b = 00007FFE3FEDAF60
x = 3, address of x = 000000000042E830
y = 18.0, address of y = 000000000042E840
```

passVstru.pas

main.pas

```
1  program EmployeeProgram;
2
3  type
4    employee = record
5      name: string[100];
6      age: integer;
7      salary: real;
8      department: string[50];
9    end;
10
11 // This procedure takes structure variable as parameter
12 procedure printEmployeeDetails(emp: employee);
13 begin
14   writeln;
15   writeln('*** Employee Details ***');
16   writeln('Name : ', emp.name);
17   writeln('Age : ', emp.age);
18   writeln('Salary : ', emp.salary:0:2);
19   writeln('Department : ', emp.department);
20 end;
21
22 var
23   manager: employee;
24
25 begin
26   // Assigning data to members of structure variable
27   write('Name: ');
28   readln(manager.name);
29   write('Age: ');
30   readln(manager.age);
31   write('Salary: ');
32   readln(manager.salary);
33   write('Department: ');
34   readln(manager.department);
35
36   // Passing structure variable to procedure
37   printEmployeeDetails(manager);
38 end.
```

Pass array(1)

Run | Debug | Stop | Share | Save | { } Beautify

ain.pas

passAry1.pas

```
1  program passArrayDemo1;
2
3  // This procedure takes structure variable as parameter
4  procedure printArray(data: array[1..5] of integer );
5  begin
6    writeln;
7    writeln('data[1] =  ', data[1]);
8    writeln('data[2] =  ', data[2]);
9    data[1] := 101;
10   data[2] := 202;
11 end;
12
13 var
14   myData2: array[1..5] of integer;
15
16 begin
17   myData2[1] := 110;  myData2[2] := 120;
18   printArray(myData2);
19   writeln('myData2[1] =  ', myData2[1]);
20   writeln('myData2[2] =  ', myData2[2]);
21
22 end.
```

```
Compiling main.pas
main.pas(4,33) Fatal: Syntax error, "OF" expected but "[" found
Fatal: Compilation aborted
Error: /usr/bin/ppcx64 returned an error exitcode
```

passAry2.pas

main.pas

```
1  program passArrayDemo2;
2
3  type
4    myIntArray = array[1..5] of integer;
5
6  // This procedure takes structure variable as parameter
7  procedure printArray(data: myIntArray);
8  begin
9    writeln;
10   writeln('data[1] =  ', data[1]);
11   writeln('data[2] =  ', data[2]);
12   data[1] := 101;
13   data[2] := 202;
14 end;
15
16 var
17   myData1: myIntArray;
18   myData2: array[1..5] of integer;
19
20 begin
21   myData1[1] := 10;  myData1[2] := 20;
22   printArray(myData1);
23   writeln('myData1[1] =  ', myData1[1]);
24   writeln('myData1[2] =  ', myData1[2]);
25   myData2[1] := 110;  myData2[2] := 120;
26   printArray(myData2);
27   writeln('myData2[1] =  ', myData2[1]);
28   writeln('myData2[2] =  ', myData2[2]);
29
30 end.
```

```
data[1] =  10
data[2] =  20
myData1[1] =  10
myData1[2] =  20

data[1] =  110
data[2] =  120
myData2[1] =  110
myData2[2] =  120
```

7.2 Passing Data

Passing by reference

Parameter : formal parameters
Argument : actual parameters

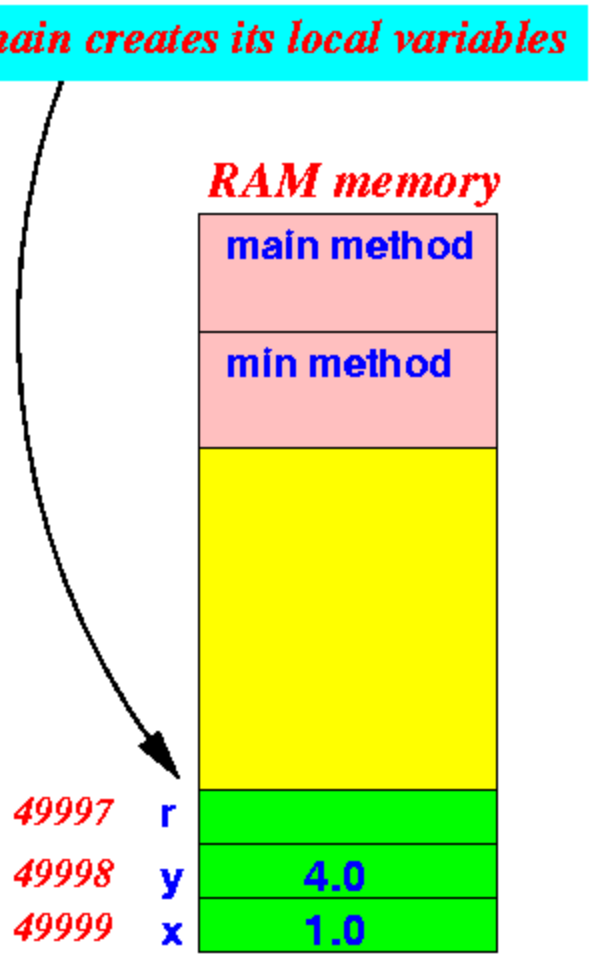
- The pass by reference is a two ways communication between the caller and its called function.
  - One for sending data from the caller to the function
  - One for receiving data of caller from its function
- However, the mainly objective of the method is getting information from other function.

ตย2. กลไกล pass by reference

Pass-by-reference mechanism

```
public class MyProgram
{
   public  static  void  main( String[] args )
   {
       double x = 1.0, y = 4.0;
       double r;
       r =  ToolBox.min( x , y );
       ....
   }
}
```

```
public class ToolBox
{
    public static double  min ( double a, double b  )
    {
        double m = 0;
        if ( a < b )
        {
          m = a;
        }
        else
        {
          m = b;
        }
        return(m);
    }
}
```

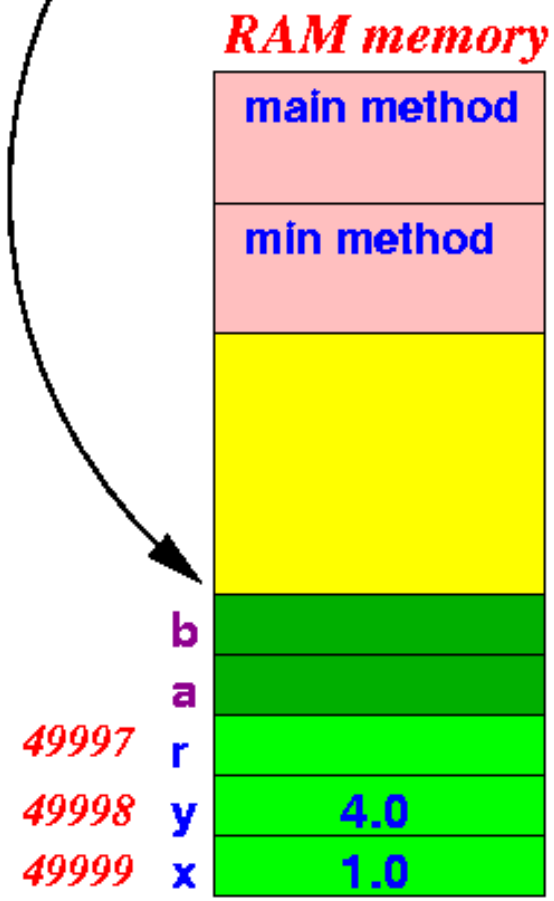


http://www.cs.emory.edu/~cheung/Courses/170/Syllabus/08/parameter3.html

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

      double x = 1.0, y = 4.0;
      double r;
      r =  ToolBox.min(  x ,  y );

      ....

   }
}
```

```
public class ToolBox
{
    public static double  min ( double a, double b  )
    {
        double m = 0;

        if ( a < b )
        {
           m = a;
        }
        else
        {
           m = b;
        }
        return(m);
    }
}
```

Pass-by-reference:
1. create parameter variables

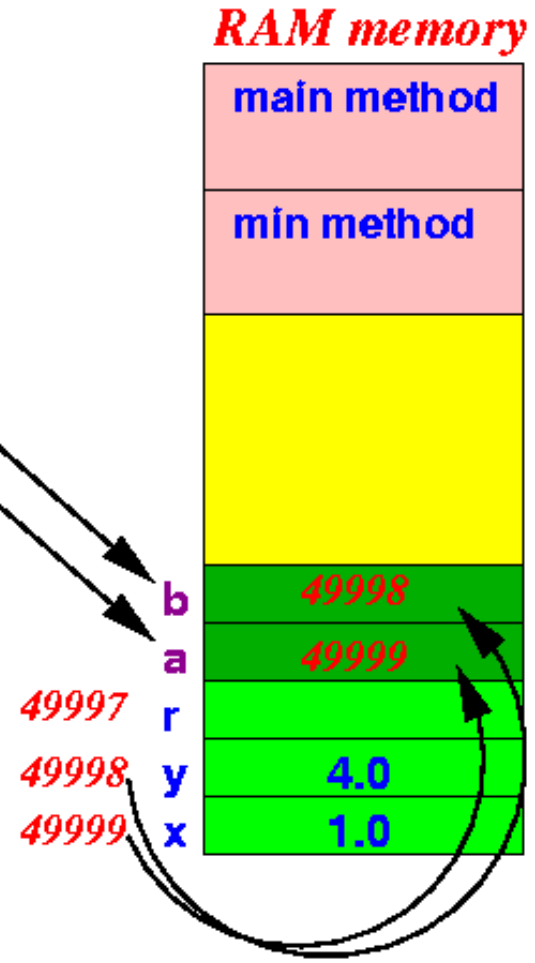


```
public class MyProgram
{
   public  static  void  main( String[] args )
   {
       double x = 1.0, y = 4.0;
       double r;
       r =  ToolBox.min( x , y );
       ....
   }
}
```

```
public class ToolBox
{
    public static double  min ( double a, double b )
    {
        double m = 0;
        if ( a < b )
        {
           m = a;
        }
        else
        {
           m = b;
        }
        return(m);
    }
}
```

Pass-by-reference:

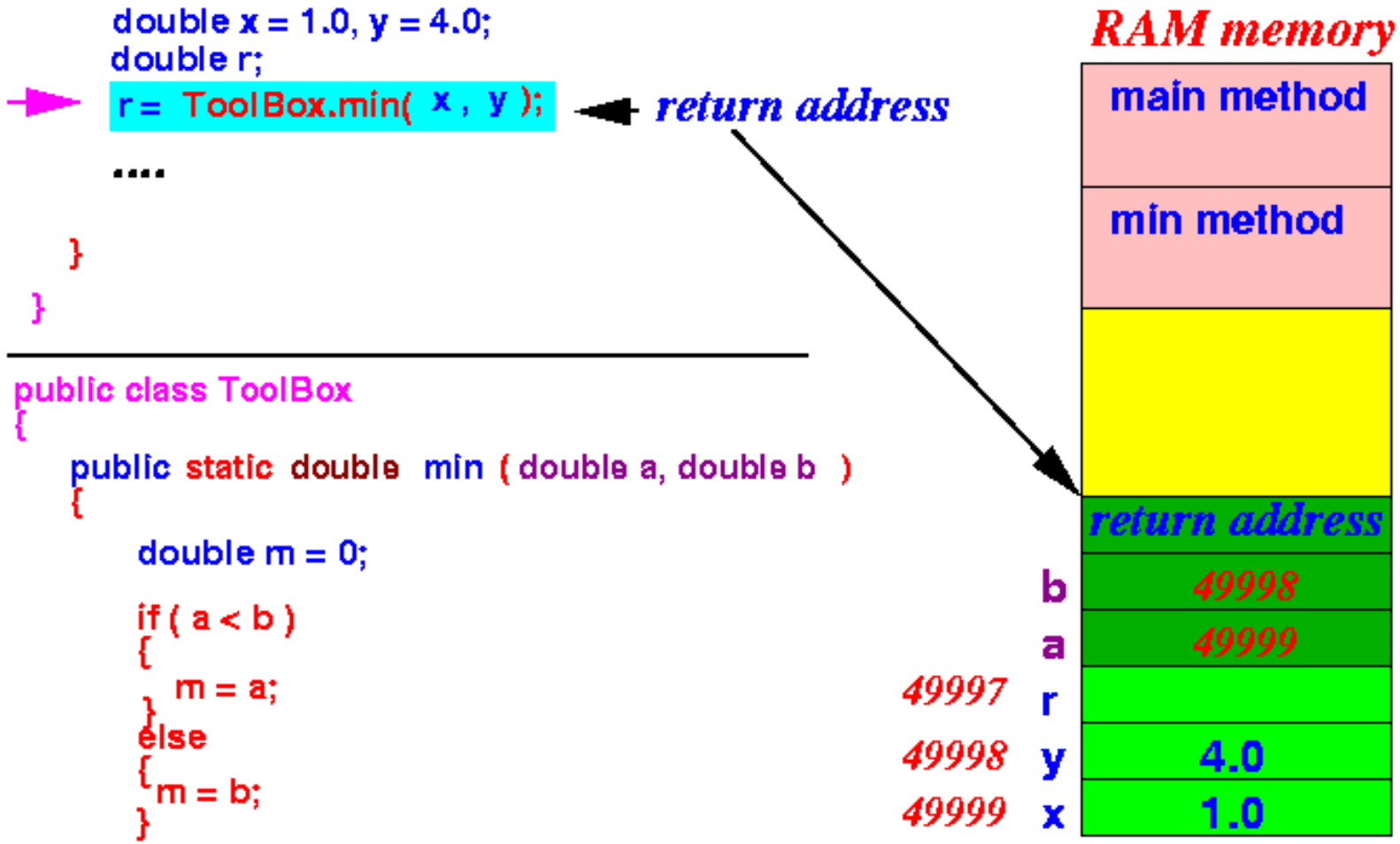
2. copy reference of actual parameter to formal parameter



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

      double x = 1.0, y = 4.0;
      double r;
      r =  ToolBox.min( x , y );

      ....

   }
}
```

```
public class ToolBox
{
    public static double  min ( double a, double b  )
    {
        double m = 0;

        if ( a < b )
        {
           m = a;
        }
        else
        {
           m = b;
        }
        return(m);
    }
}
```

Method invocation mechanism: Save return address



Method invocation mechanism: Jump to method

```
public class MyProgram
{
   public  static  void  main( String[] args )
   {
       double x = 1.0, y = 4.0;
       double r;
       r =  ToolBox.min( x , y );
       ....
   }
}
```

return address

```
public class ToolBox
{
    public static double  min ( double a, double b  )
    {
        double m = 0;

        if ( a < b )
        {
           m = a;
        }
        else
        {
           m = b;
        }
        return(m);
    }
}
```

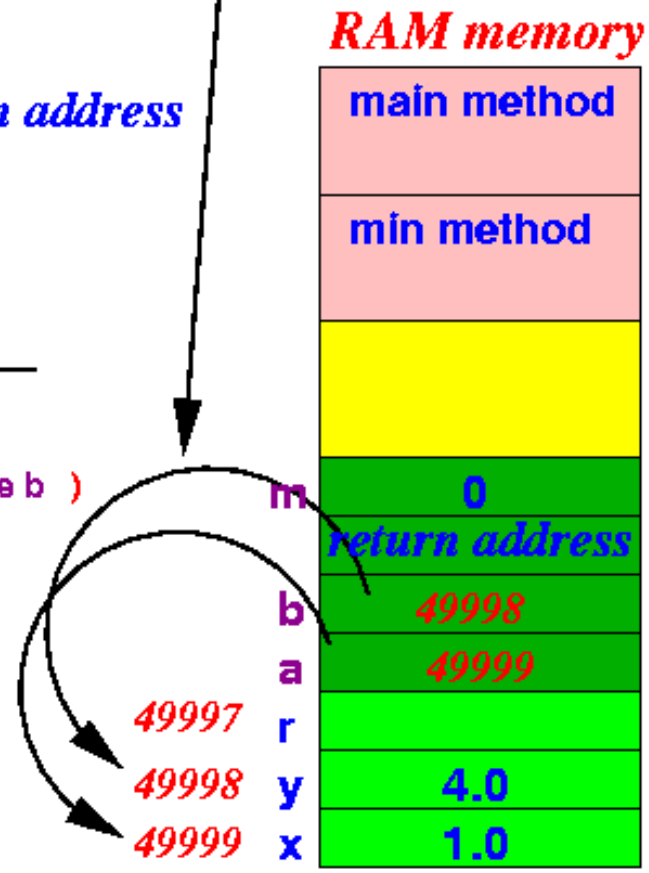
RAM memory

```
public class MyProgram
{
   public  static  void  main ( String[] args )
   {
       double x = 1.0, y = 4.0;
       double r;
       r =  ToolBox.min( x , y );
       ....
   }
}
```

return address

Method access information by:
1. Locate the actual parameter

```
public class ToolBox
{
    public static double  min ( double a, double b  )
    {
        double m = 0;
        if ( a < b )
        {
           m = a;
        }
        else
        {
           m = b;
        }
        return(m);
    }
}
```



```
public class MyProgram
{
   public  static  void  main ( String[] args )
   {
      double x = 1.0, y = 4.0;
      double r;
      r =  ToolBox.min( x , y );
      ....
   }
}
```

return address

Method access information by:
2. Extract the information from the actual parameters

```
public class ToolBox
{
    public static double  min ( double a, double b  )
    {
        double m = 0;
        if ( a < b )
        {
           m = a;
        }
        else
        {
           m = b;
        }
        return(m);
    }
}
```

RAM memory

	main method	
	min method	
m	0	
	return address	
b	*49998*	
a	*49999*	
r		*49997*
y	4.0	*49998*
x	1.0	*49999*

Pass-by-reference mechanism update value case

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

       double x = 1.0, y = 4.0;
       double r;
       r =  ToolBox.fun(  x ,  y );

       System.out.println(x);
       System.out.println(y);
       System.out.println(r);
   }
}
```

```
public class ToolBox
{
    public static double  fun  ( double a, double b  )
    {
        double m = 0;
        a = a + 1;
        b = b + 2;
        m = a + b;
        return(m);

    }
}
```

Situation when method **fun** ***starts running***

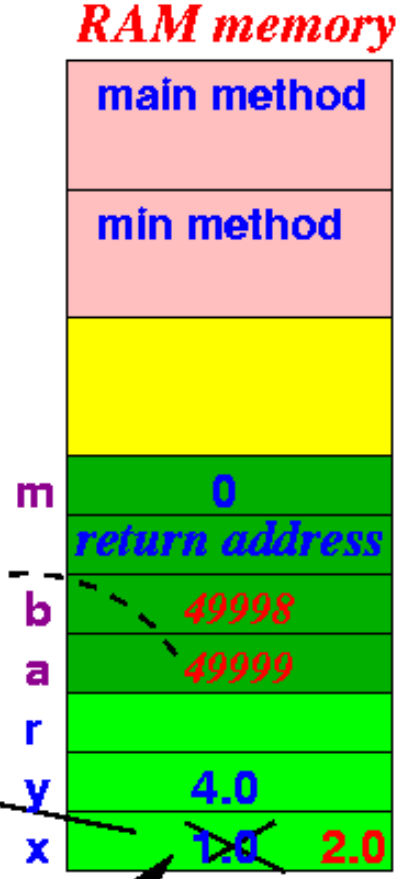
RAM memory

		main method
		min method
	m	0
		return address
	b	*49998*
	a	*49999*
49997	r	
49998	y	4.0
49999	x	1.0

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

       double x = 1.0, y = 4.0;
       double r;
       r =  ToolBox.fun(  x ,  y );

       System.out.println(x);
       System.out.println(y);
       System.out.println(r);
   }
}
```

```
public class ToolBox
{
    public static double  fun  ( double a, double b  )
    {
        double m = 0;
        a = a + 1;
        b = b + 2;
        m = a + b;
        return(m);

    }
}
```

Assignment statements changes values stored in actual parameter



Address parameter vs argument

main.pas

passRefDm1.pas

```
1  program PassByRef;
2
3  procedure PassRef(a: integer; var b: real);
4  begin
5      a := a * 2;
6      b := b + 2;
7      writeln('a = ', a , ', address of a = ' , HexStr(@a));
8      writeln('b = ', b:0:1 , ', address of b = ' , HexStr( @b ));
9  end;
10
11 var
12     x: integer;
13     y: real;
14
15 begin
16     x := 3;
17     y := 18.0;
18     writeln('1.x = ', x , ', address of x = ' , HexStr(@x));
19     writeln('2.y = ', y:0:1 , ', address of y = ' , HexStr( @y ));
20     PassRef(x, y);
21     writeln('3.x = ', x , ', address of x = ' , HexStr(@x));
22     writeln('4.y = ', y:0:1 , ', address of y = ' , HexStr( @y ));
23 end.
24
```

```
1.x = 3, address of x = 000000000042E830
2.y = 18.0, address of y = 000000000042E840
a = 6, address of a = 00007FFF20DF99A8
b = 20.0, address of b = 000000000042E840
3.x = 3, address of x = 000000000042E830
4.y = 20.0, address of y = 000000000042E840
```

passStru.pas

ตย.การ pass structure

main.pas

Ctrl+S

```
1  program SwapStructExample;
2
3  type
4    TMyStruct = record
5      x: Integer;
6      y: Integer;
7    end;
8
9  procedure Swap(var ms: TMyStruct);
10 var
11   temp: Integer;
12 begin
13   temp := ms.x;
14   ms.x := ms.y;
15   ms.y := temp;
16 end;
17
18 var
19   ms: TMyStruct;
20 begin
21   ms.x := 50;
22   ms.y := 60;
23
24   WriteLn;
25   WriteLn('Before swap in main:');
26   WriteLn('x = ', ms.x, #9, 'y = ', ms.y);
27
28   Swap(ms);
29
30   WriteLn;
31   WriteLn('After swap in main:');
32   WriteLn('x = ', ms.x, #9, 'y = ', ms.y);
33 end.
```

```
Before swap in main:
x = 50   y = 60

After swap in main:
x = 60   y = 50
```

Run | Debug | Stop | Share | Save | { } Beautify

main.pas

```pascal
program passArrayRef;

type
  myIntArray = array[1..5] of integer;

// This procedure takes structure variable as parameter
procedure printArray(var data: myIntArray);
begin
  writeln;
  writeln(chr(9)+'data[1] =  ', data[1]);
  writeln(chr(9)+'data[2] =  ', data[2]);
  data[1] := 101;
  data[2] := 202;
end;

var
  myData1: myIntArray;
  myData2: array[1..5] of integer;

begin
  myData1[1] := 10;  myData1[2] := 20;
  printArray(myData1);
  writeln('myData1[1] =  ', myData1[1]);
  writeln('myData1[2] =  ', myData1[2]);
  myData2[1] := 110;  myData2[2] := 120;
  printArray(myData2);
  writeln('myData2[1] =  ', myData2[1]);
  writeln('myData2[2] =  ', myData2[2]);

end.
```

```
        data[1] =  10
        data[2] =  20
myData1[1] =  101
myData1[2] =  202

        data[1] =  110
        data[2] =  120
myData2[1] =  101
myData2[2] =  202
```

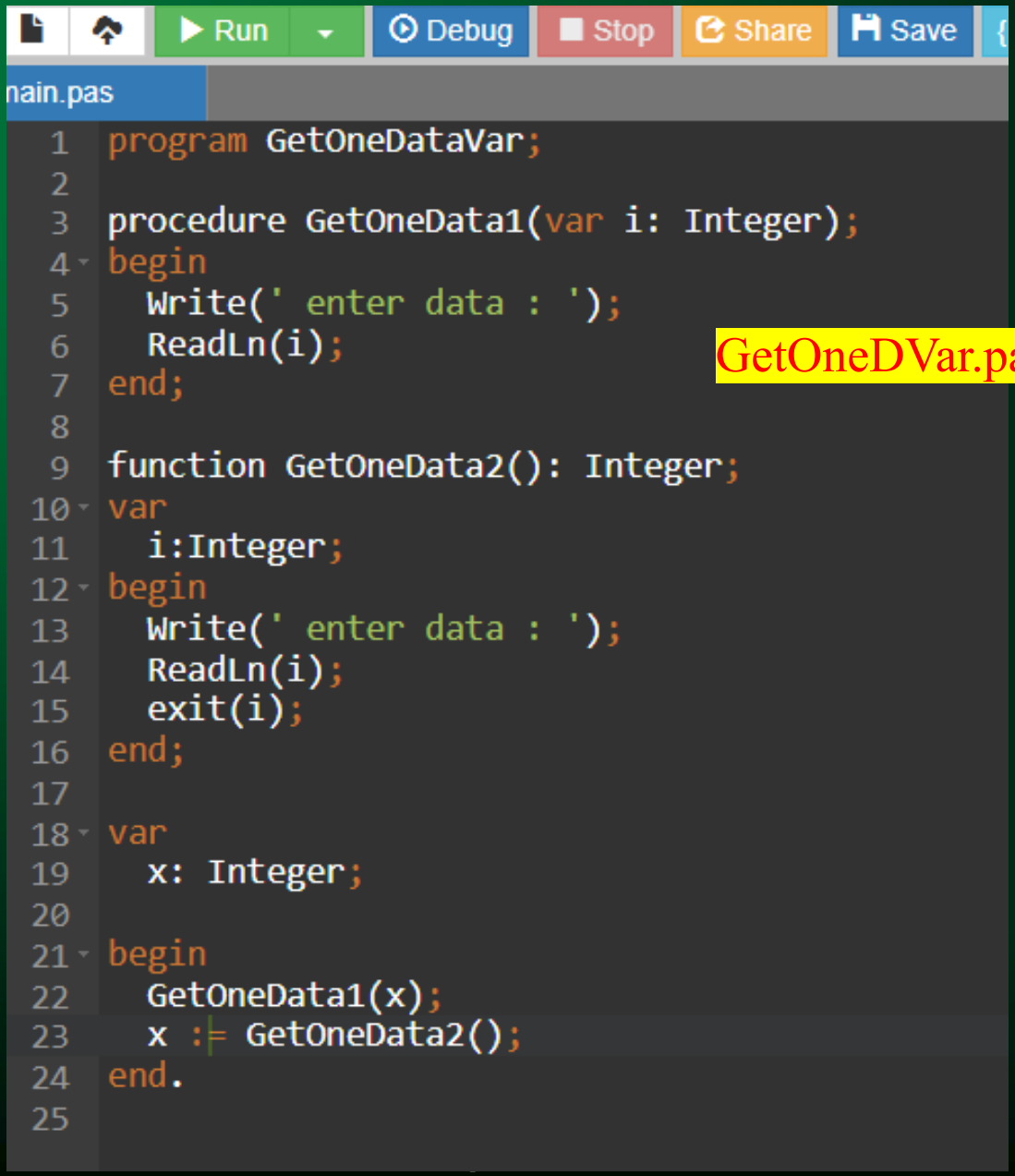
ตย.การpass array

passAryRef.pas

Apirak :

แนวทางการออกแบบ subrountine

Get single value



GetOneDVar.pas

Return multiple value using pass by reference

main.pas

```
1  program DoSomething4Example;
2
3  function DoSomething1(var i, j: Integer): Integer;
4  var
5    k: Integer;
6  begin
7    ReadLn(i);
8    ReadLn(j);
9    ReadLn(k);
10   exit(k);
11 end;
12
13 function DoSomething2(var i, j ,k :Integer): Integer;
14 begin
15   ReadLn(i);
16   ReadLn(j);
17   ReadLn(k);
18   exit(0); // 0 -> status of function
19 end;
20
21 procedure DoSomething3(var i, j ,k :Integer;
22 begin
23   ReadLn(i);
24   ReadLn(j);
25   ReadLn(k);
26 end;
27
28 var
29   a, b, c: Integer;
30 begin
31   c := DoSomething1(a, b);
32   DoSomething2(a, b,c);
33   DoSomething3(a, b,c);
34 end.
35
```

multRetV.pas

Run | Debug | Stop | Share | Sav

main.pas

```pascal
program arrayGuide;
const
    n = 3;
type
    vector= array[1..n] of Real;

function GetData():vector;
var
  Z: vector;
begin
  :
  exit(z);
end;

procedure ReadData(var Z: vector);
begin
  :
end;

function Sum( Z: vector): Real;
var
  s: Real;
begin
  s := 0;
  :
  exit(s);
end;
```

```pascal

function Mult(var Z: vector): Real;
var
  s: Real;
  C : array [1..3] of integer;
  E : vector;
begin
  s := 0;
  :
  exit(s);
end;

var
  A, B : vector;
  A_bar,B_bar : real;
  C : array [1..3] of integer;
begin
  WriteLn('Input DATA of Matrix A');
  A := GetData();
  WriteLn('Input DATA of Matrix B');
  ReadData( B );          //** best
  A_bar := Sum(A);
  B_bar := Mult(B);       //** best

end.
```

arrayGuide.pas

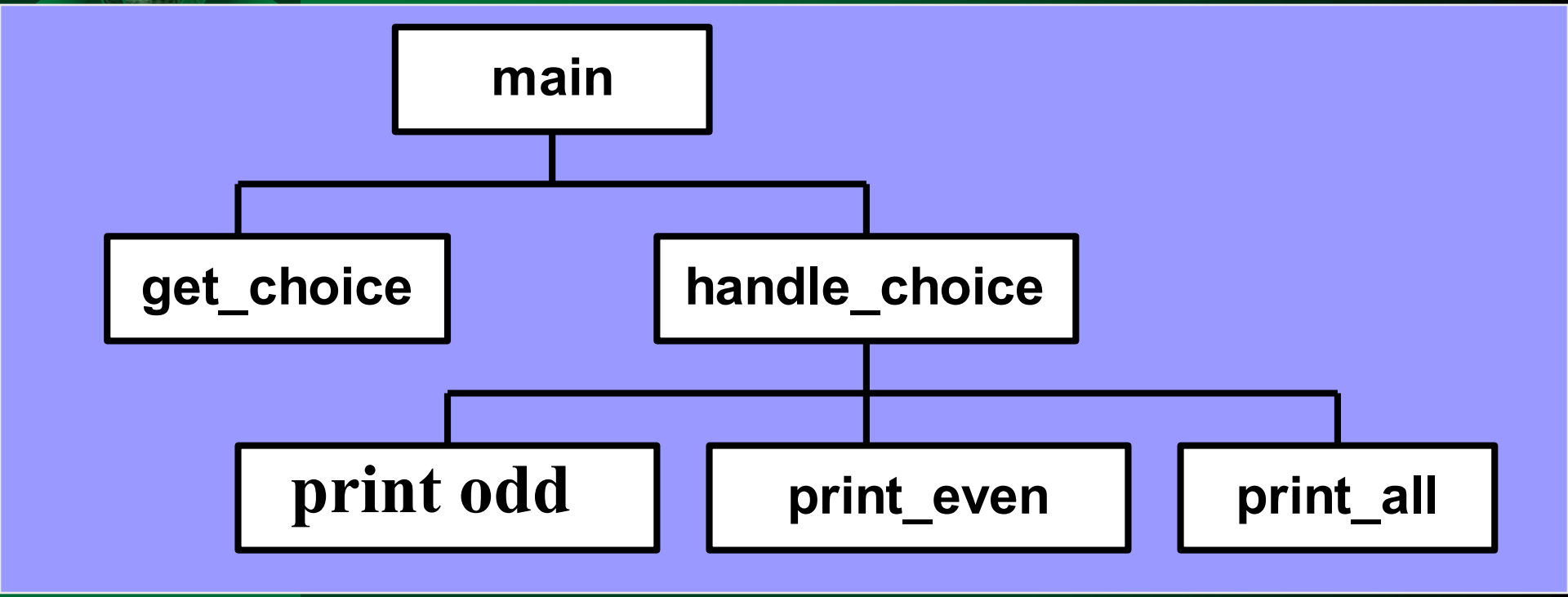
แนวปฏิบัติในการเขียน code ของ pascal ที่เกี่ยวกับ array

Apirak :

Apirak :

ตัวอย่างโปรแกรม

Rewrite function using pass by reference



- rewrite a program by dividing activities to correspond with about diagram.

main.pas

```
1  program NumberSeries;
2  var
3    choice, i: integer;
4
5  begin
6
7    repeat
8      writeln('Which series do you wish to display?');
9      writeln('1 - Odd numbers from 1 to 30');
10     writeln('2 - Even numbers from 1 to 30');
11     writeln('3 - All numbers from 1 to 30');
12     write('Enter your choice (1-3): ');
13     readln(choice);
14
15     if (choice < 1) or (choice > 3) then
16       writeln('Choice must be 1, 2, or 3');
17
18   until (choice >= 1) and (choice <= 3);
19
20   case choice of
21     1: begin
22          for i := 1 to 30 do
23            if (i mod 2 <> 0) then
24              write(i, ' ');
25          writeln;
26        end;
27     2: begin
28          for i := 1 to 30 do
29            if (i mod 2 = 0) then
30              write(i, ' ');
31          writeln;
32        end;
33     3: begin
34          for i := 1 to 30 do
35            write(i, ' ');
36          writeln;
37        end;
38   end;
39
40 end.
```

main.pas

NumSeriV2.pas

```
1  program NumberSeriesMenu;
2
3  function GetChoice(): Integer;
4  var
5    choice: Integer;
6  begin
7    repeat
8      WriteLn('Which series do you wish to display?');
9      WriteLn('1 - Odd numbers from 1 to 30');
10     WriteLn('2 - Even numbers from 1 to 30');
11     WriteLn('3 - All numbers from 1 to 30');
12     Write('Enter choice (1-3): ');
13     ReadLn(choice);
14     if (choice < 1) or (choice > 3) then
15       WriteLn('Choice must be 1, 2, or 3');
16   until (choice >= 1) and (choice <= 3);
17   Result := choice;
18 end;
19
20 procedure PrintOdd;
21 var
22   i: Integer;
23 begin
24   for i := 1 to 30 do
25     if (i mod 2 = 1) then
26       Write(i, ' ');
27   WriteLn;
28 end;
29
```

NumSeriV2.pas

main.pas

```
30  procedure PrintEven;
31  var
32    i: Integer;
33  begin
34    for i := 1 to 30 do
35      if (i mod 2 = 0) then
36        Write(i, ' ');
37    WriteLn;
38  end;
39
40  procedure PrintAll;
41  var
42    i: Integer;
43  begin
44    for i := 1 to 30 do
45      Write(i, ' ');
46    WriteLn;
47  end;
48
49  var
50    choice: Integer;
51  begin
52    choice := GetChoice;
53
54    case choice of
55      1: PrintOdd;
56      2: PrintEven;
57      3: PrintAll;
58    end;
59
60    WriteLn;
61    Write('Press <Enter> to exit...');
62    ReadLn;
63  end.
```

```
48
49  var
50    choice: Integer;
51  begin
52    choice := GetChoice;
53
54    case choice of
55      1: PrintOdd;
56      2: PrintEven;
57      3: PrintAll;
58    end;
59
60    WriteLn;
61    Write('Press <Enter> to exit...');
62    ReadLn;
63  end.
```

Aplikasi

Rewrite program with procedural concept

แบบฝึกหัด 2

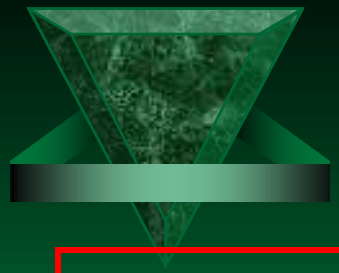
จงเขียนโปรแกรมต้องการนำระเบียนลูกค้า(customer) ของสหกรณ์ออมทรัพย์รามคำแหง ที่รองรับการจัดเก็บข้อมูลไม่เกิน 500 คน ระเบียนลูกค้าประกอบด้วย รหัสบัญชี(id) ชื่อบัญชี(name) ที่อยู่(Address) และเงินฝาก (Deposit)

Interest = deposit*10%;
Deposist = Deposit+interst

Menu

1. Open a new Account
2. Deposit
3. Withdraw
4. Compound interest
5. Quit

Please select 1 / 2 / 3 / 4 / 5 =?



main
Opennewacc
widthdraw
report
menu
deposit
Compond interest

main.pas

```
program BankAccount;

type
  Account = record
    id: string[10];
    name: string[100];
    addr: string[100];
    deposit: real;
  end;

var
  cust: array[1..500] of Account;
  n, choice, i: integer;
  id: string[10];
  bfound: boolean;
  deposit: real;

begin
  n := 0;
  repeat
    writeln;
    writeln('1. Open Acc');
    writeln('2. Deposit Acc');
    writeln('3. Withdraw Acc');
    writeln('4. Compute Interest');
    writeln('5. Exit');
    write('Enter choice: ');
    readln(choice);
```

```
    case choice of
      1:
        begin
          writeln;
          writeln('Do Open Acc');
          write('id Acc =? ');
          readln(id);
          bfound := False;

          for i := 1 to n do
          begin
            if cust[i].id = id then
            begin
              bfound := True;
              break;
            end; // end if
          end; //end for

          if  bfound = False then
            begin
                n := n + 1;
                cust[n].id := id;
                write('name Acc =? ');
                readln(cust[n].name);
                write('addr Acc =? ');
                readln(cust[n].addr);
                write('deposit Acc =? ');
                readln(cust[n].deposit);
            end
          else
            writeln('id Acc found = ', id);

        end; //case 1
```

Single module

Apirak :

main.pas

```
1  program BankAccount;
2
3  type
4    Account = record
5      id: string[10];
6      name: string[100];
7      addr: string[100];
8      deposit: real;
9    end;
10
11   AccountDataType = array[1..500] of Account;
12
13 function getChoice:integer;
14 var
15   choice:integer;
16 begin
17     writeln;
18     writeln('1. Open Acc');
19     writeln('2. Deposit Acc');
20     writeln('3. Withdraw Acc');
21     writeln('4. Compute Interest');
22     writeln('5. Exit');
23     write('Enter choice: ');
24     readln(choice);
25     exit(choice);
26 end;
```

BackAccV2.pas

BackAccV2.pas

```
28 Procedure OpenAcc( var cust:AccountDataType ; var n:integer );
29 var
30       id: string[10];
31       bfound: boolean;
32       i:integer;
33 begin
34     writeln;
35     writeln('Do Open Acc');
36     write('id Acc =? ');
37     readln(id);
38     bfound := False;
39
40     for i := 1 to n do
41     begin
42     if cust[i].id = id then
43     begin
44       bfound := True;
45       break;
46     end; // end if
47     end; //end for
48
49     if  bfound = False then
50     begin
51         n := n + 1;
52         cust[n].id := id;
53         write('name Acc =? ');
54         readln(cust[n].name);
55         write('addr Acc =? ');
56         readln(cust[n].addr);
57         write('deposit Acc =? ');
58         readln(cust[n].deposit);
59     end
60     else
61         writeln('id Acc found = ', id);
62 end;
```

BackAccV2.pas

```
64  Procedure Deposit( var cust:AccountDataType ; var n:integer );
65  var
66        id: string[10];
67        bfound: boolean;
68        deposit: real;
69        i:integer;
70  begin
71    writeln;
72    writeln('Do Deposit Acc');
73    write('id Acc =? ');
74    readln(id);
75
76    bfound := False;
77    for i := 1 to n do
78    begin
79      if cust[i].id = id then
80      begin
81        bfound := True;
82        break;
83      end; //end if
84    end; // end for
85
86    if bfound = False then
87      writeln('id Acc not found = ', id)
88    else
89     begin
90        writeln('Before deposit Acc = ', cust[i].deposit:0:2);
91        write('New deposit Acc =? ');
92        readln(deposit);
93        cust[i].deposit := cust[i].deposit + deposit;
94        writeln('After deposit Acc = ', cust[i].deposit:0:2);
95     end;
96  end; //case 2
```

```
98
99  var
100   cust: AccountDataType;
101   n, choice, i: integer;
102
103
104 begin
105   n := 0;
106   repeat
107     choice :=  getChoice();
108     case choice of
109       1: OpenAcc( cust , n );
110       2: Deposit( cust ,n );
111       3:
112         begin
113           writeln('Do Withdraw Acc');
114         end;
115       4:
116         begin
117           writeln('Do Interest Acc');
118         end;
119     end;
120
121   until choice = 5;
122 end.
123
```

BackAccV2.pas

```
1. Open Acc
2. Deposit Acc
3. Withdraw Acc
4. Compute Interest
5. Exit
Enter choice: 2

Do Deposit Acc
id Acc =? i003
id Acc not found = i003

1. Open Acc
2. Deposit Acc
3. Withdraw Acc
4. Compute Interest
5. Exit
Enter choice: 2

Do Deposit Acc
id Acc =? i001
Before deposit Acc = 330.00
New deposit Acc =? 20
After deposit Acc = 350.00

1. Open Acc
2. Deposit Acc
3. Withdraw Acc
4. Compute Interest
5. Exit
Enter choice: 5
```

ตัวอย่าง

ให้นักศึกษาเขียนโปรแกรมขายสินค้าและพิมพ์ใบเสร็จรับเงิน
โดยข้อมูลในระบบมีไม่เกิน **100** รายการ
ข้อมูลขายสินค้าด้วย

- รหัสสินค้า(id) เช่น “AB001”
- ชื่อสินค้า(name) เช่น “CAFÉ”
- ราคาขาย(cost) เช่น 30.6
- จำนวนสินค้าในสต๊อก (instock) เช่น 5

```
        Menu
1.      Add item to Cart
2.      Sell Product
3.      Check Stock
4.      End Program
Please select 1 / 2 / 3/ 4 =?
```

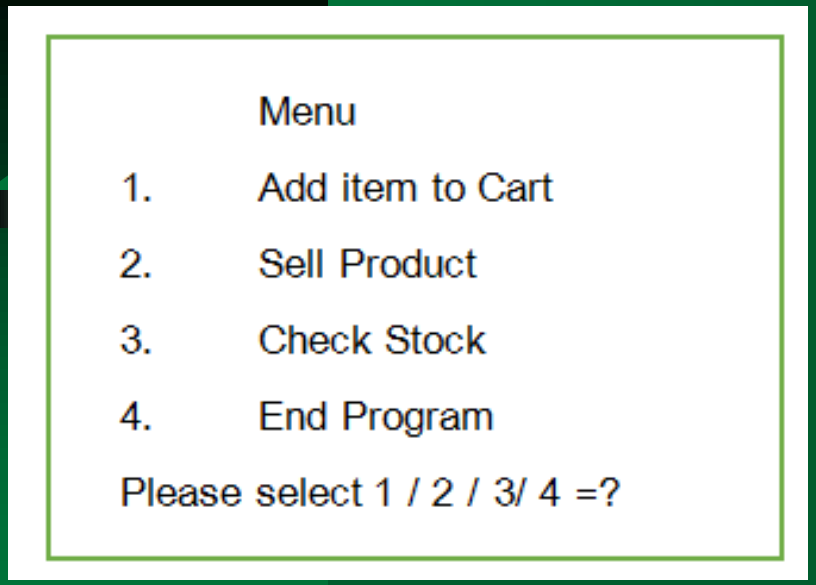
Rewrite your code with Function approach

Rewrite the code using function approach

ตัวอย่าง

ให้นักศึกษาเขียนโปรแกรมขายสินค้าและพิมพ์ใบเสร็จรับเงิน
โดยข้อมูลในระบบมีไม่เกิน 100 รายการ
ข้อมูลขายสินค้าด้วย

- รหัสสินค้า(id) เช่น “AB001”
- ชื่อสินค้า(name) เช่น “CAFÉ”
- ราคาขาย(cost) เช่น 30.6
- จำนวนสินค้าในสต๊อก (instock) เช่น 5

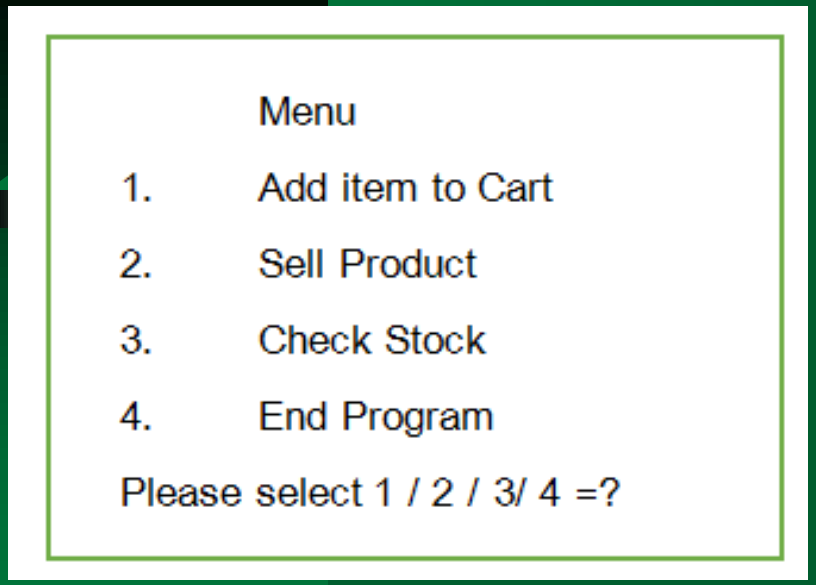


กด 1

Input your product data:
id : AB001
name : CAFE
Cost : 30.6
Stock : 5

กด 2

Input your product id: AB001
Input your sell volume : 2
name : CAFE
Cost : 30.6
stock : 3
Pay : 61.2 bath



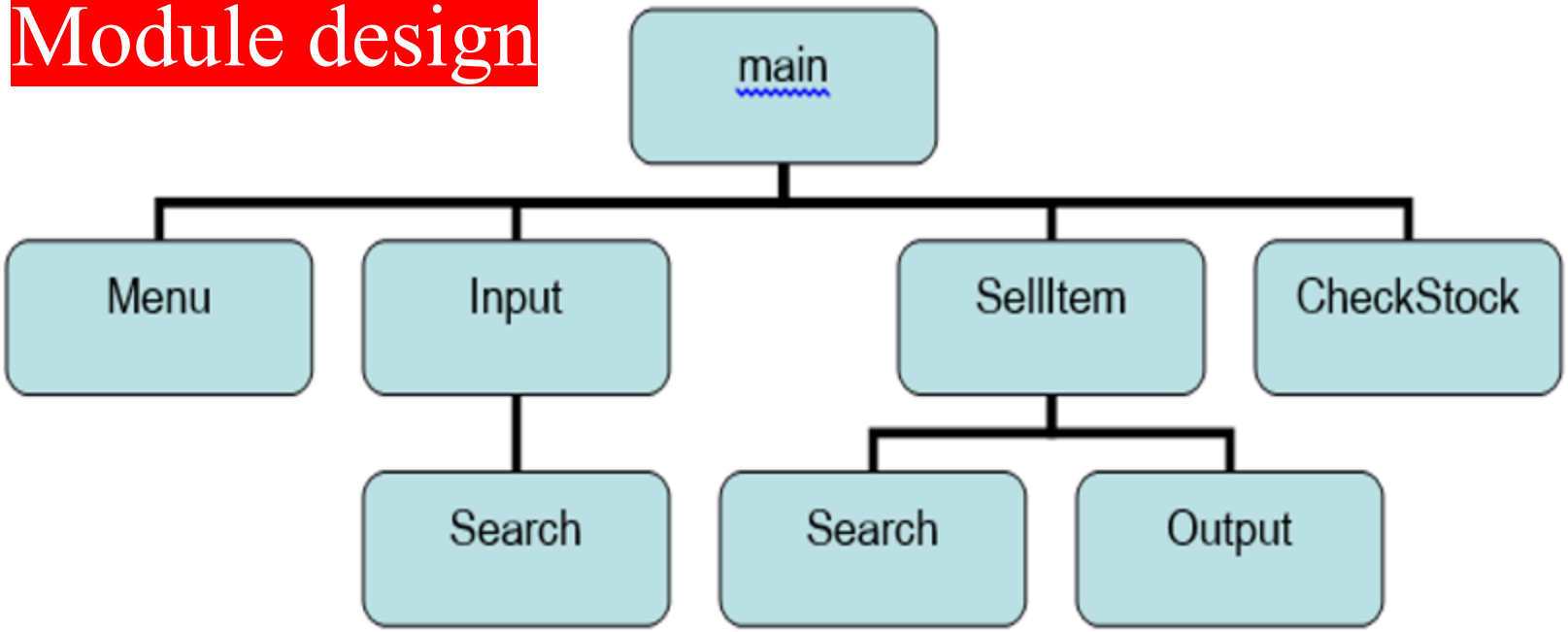
กด 3

```
product List:
Id          name    cost  stock
- - - - - - - - - - - - - - - - - -
AB001  CAFÉ   30.6    3
```

กด 4

exit from program

Module design



EX1 จากตัวอย่างข้างต้นให้ นักศึกษาปรับโปรแกรม โดยให้มีการตรวจดังนี้

กรณี กด 1 : ให้ตรวจสอบว่ารหัสที่ป้อน มีซ้ำหรือไม่ ถ้าไม่มีจึงจะอนุญาตให้ผู้ใช้ป้อนข้อมูล

กรณี กด 2: ให้ตรวจสอบว่ารหัสสินค้าที่ผู้ใช้ป้อนมีอยู่ในระบบหรือไม่ กรณีไม่พบให้แสดงข้อความ “product not exist”

กรณี กด 3 ให้แสดงรายการทั้งหมด และแสดงยอดรวมของ cost

Design methodology

- ***Top-down design**: begins with the function at top of diagram and works toward the function at the bottom of the diagram.*
- ***Button-up design**: involves begins with the bottom of diagram and working your way up.*

ตัวอย่าง (3)

จงเขียนโปรแกรมอ่านข้อมูล 2 ชนิด คือ A และ B ซึ่งชนิดละ20 จำนวน จากนั้นให้คำนวณค่า

A_bar = Sum i = 1 .. 20 ของ A

B_bar = Sum i = 1 .. 20 ของ B

$$Corr(A, B) = \frac{n*Sum(AiBi) - Sum(Ai)*Sum(Bi)}{sqrt(n*(sum(Ai*Ai) - Sum(Ai)*Sum(Ai)) * n*(sum(Bi*Bi) - Sum(Bi)*Sum(Bi)))}$$

กำหนดให้ z = sqrt(x) เป็นbuild in function ของการหาค่า root x

CorrelationAB.pas

ain.pas

```pascal
program CorrelationAB;
const
    n = 3;

type
    vector= array[1..n] of Real;

function GetData():vector;
var
  i: Integer;
  Z: vector;
begin
  for i := 0 to n - 1 do
  begin
    Write('Enter value [', i + 1, ']: ');
    ReadLn(Z[i]);
  end;
  exit(z);
end;

function Sum( Z: vector): Real;
var
  s: Real;
  i: Integer;
begin
  s := 0;
  for i := 1 to n  do
    s := s + Z[i];
  exit(s);
end;

function MultAry( W, X:vector): vector;
var
  i: Integer;
  Z: vector;
begin
  for i := 1 to n  do
    Z[i] := W[i] * X[i];
  exit(Z);
end;
```

```
41
42  var
43    A, B, T: vector;
44    A_bar, B_bar, corrAB, X, Y: Real;
45    AB_bar, AA_bar, BB_bar: Real;
46
47  begin
48    WriteLn('Input DATA of Matrix A');
49    A := GetData();
50    WriteLn('Input DATA of Matrix B');
51    B := GetData();
52
53    A_bar := Sum(A);
54    B_bar := Sum(B);
55
56    T := MultAry(A, A );
57    AA_bar := Sum(T);
58
59    T := MultAry(B, B);
60    BB_bar := Sum(T);
61
62    T := MultAry(A, B);
63    AB_bar := Sum(T);
64
65    X := (n * AB_bar - A_bar * B_bar);
66    Y := (n * (AA_bar - A_bar * A_bar)) * (n * (BB_bar - B_bar * B_bar));
67
68    corrAB := X / Sqrt(Y);
69
70    WriteLn;
71    WriteLn('corr(A,B) = ', corrAB:0:6);
72
73    WriteLn;
74    Write('Press <Enter> to exit...');
75    ReadLn;
76  end.
```

Apirak :

สมมุติว่า Apartment แห่งหนึ่ง มีจำนวนห้องพัก 100 ห้อง จงเขียนโปรแกรมคำนวณค่าปริมาณการใช้น้ำดังนี้

1. โปรแกรมแสดงเมนู
    1. add meter
    2.Caculate meter
    3.Diplay meter
2. ข้อมูลที่รับเข้าประกอบด้วย

1.เลขประจำมิเตอร์ (no. of meter) จำนวน 4 ตัวอักษร เช่น A001,A002

2. ค่า start มิเตอร์ (start meter value)

3.ค่า end มิเตอร์

ตัวอย่าง แบบ top down

จงเขียนโปรแกรมรับข้อมูลลงอาเรย์ A , B ขนาด 20x20 และหาผลคูณของอาเรย์ C โดย

$$C = A \times B$$

สูตรการคูณ คือ $C_{ij} = \sum a_{ik} b_{kj}$

เมื่อ $1 <= i <= 20$ และ $1 <= j <= 20$

Break down the problem

1. Get data both A and B
2. Calculate C = A*B;
3. Show C

main.pas

```pascal
program MatrixMultiply;

const
  n = 3;
type
  Matrix = array[1..n, 1..n] of Integer;

procedure GetData(var Z: Matrix);
var
  i, j: Integer;
begin
  Randomize;
  for i := 1 to n do
  begin
    for j := 1 to n do
    begin
      { In real use, you could read from keyboard:
        Write('Row ', i, ' Col ', j, ': ');
        ReadLn(Z[i, j]);
      }
      Z[i, j] := Random(10);  { random number 0-9 }
      Write(Z[i, j]:5);
    end;
    Writeln;
  end;
end;

procedure MultiplyMatrix(var W, X: Matrix; var Z: Matrix);
var
  i, j, k, sum: Integer;
begin
  for i := 1 to n do
  begin
    for j := 1 to n do
    begin
      sum := 0;
      for k := 1 to n do
        sum := sum + W[i, k] * X[k, j];
      Z[i, j] := sum;
    end;
  end;
end;
```

```pascal
procedure PrintMatrix(var Z: Matrix);
var
  i, j: Integer;
begin
  Writeln('Value of Matrix C:');
  for i := 1 to n do
  begin
    for j := 1 to n do
      Write(Z[i, j]:5);
    Writeln;
  end;
end;

var
  A, B, C: Matrix;

begin
  Writeln('Input DATA of Matrix A:');
  GetData(A);

  Writeln('Input DATA of Matrix B:');
  GetData(B);

  MultiplyMatrix(A, B, C);

  PrintMatrix(C);
end.
```

MxM2.pas

MxM2.pas

main.pas

```
1  program MatrixMultiply;
2
3  const
4    n = 3;
5  type
6    Matrix = array[1..n, 1..n] of Integer;
7
8  procedure GetData(var Z: Matrix);
9  var
10   i, j: Integer;
11 begin
12   Randomize;
13   for i := 1 to n do
14   begin
15     for j := 1 to n do
16     begin
17       { In real use, you could read from keyboard:
18         Write('Row ', i, ' Col ', j, ': ');
19         ReadLn(Z[i, j]);
20       }
21       Z[i, j] := Random(10);  { random number 0-9 }
22       Write(Z[i, j]:5);
23     end;
24     Writeln;
25   end;
26 end;
```

```
28  procedure MultiplyMatrix(var W, X: Matrix; var Z: Matrix);
29  var
30    i, j, k, sum: Integer;
31  begin
32    for i := 1 to n do
33    begin
34      for j := 1 to n do
35      begin
36        sum := 0;
37        for k := 1 to n do
38          sum := sum + W[i, k] * X[k, j];
39        Z[i, j] := sum;
40      end;
41    end;
42  end;
43
```

$$C_{ij} = \sum a_{ik} b_{kj}$$

MxM2.pas

```
43
44  procedure PrintMatrix(var Z: Matrix);
45  var
46    i, j: Integer;
47  begin
48    Writeln('Value of Matrix C:');
49    for i := 1 to n do
50    begin
51      for j := 1 to n do
52        Write(Z[i, j]:5);
53      Writeln;
54    end;
55  end;
56
57  var
58    A, B, C: Matrix;
59
60  begin
61    Writeln('Input DATA of Matrix A:');
62    GetData(A);
63
64    Writeln('Input DATA of Matrix B:');
65    GetData(B);
66
67    MultiplyMatrix(A, B, C);
68
69    PrintMatrix(C);
70  end.
```

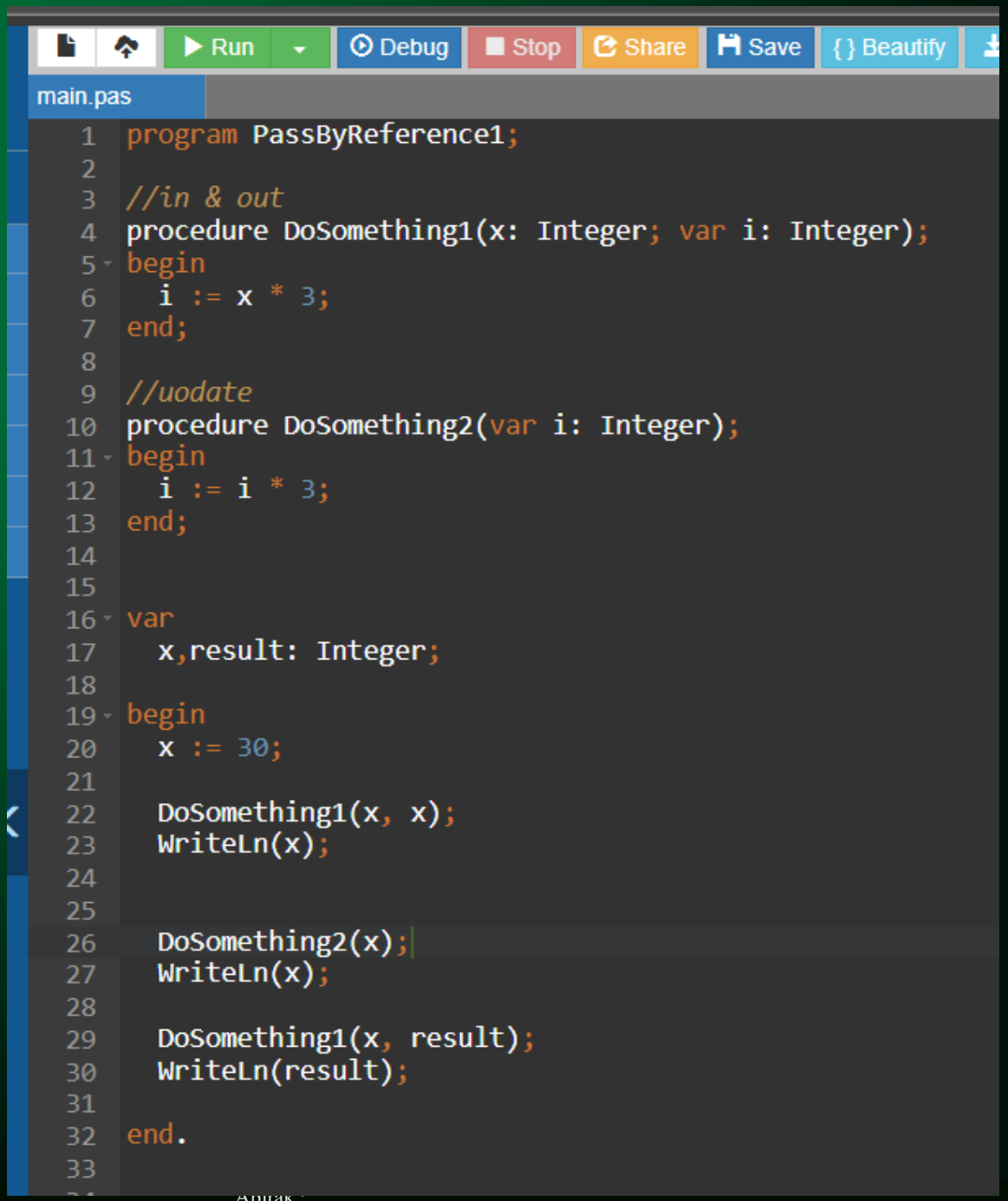
MxM2.pas

https://www.w3resource.com/cpp-exercises

END

Pass value and get value

Pass-getval.pas



```
1  program PassByValueAndReference2;
2
3  procedure pass_val(l: Integer; var a: Double);
4  begin
5    writeln(chr(9)+'pass_val before -> l = ', l);
6    writeln(chr(9)+'pass_val before -> a = ', a:0:1);
7
8    l := l * 2;
9    a := a + 2;
10
11   writeln(chr(9)+'pass_val -> l = ', l);
12   writeln(chr(9)+'pass_val -> a = ', a:0:1);
13 end;
14
15 var
16   x, i: Integer;
17   y, j: Double;
18
19 begin
20   y := 18.0;   x := 3;
21   i := 100;    j := 91.1;
22
23   writeln('main before -> x = ', x);
24   writeln('main before -> y = ', y:0:1);
25
26   pass_val(x, y);
27
28   writeln('main -> x = ', x);
29   writeln('main -> y = ', y:0:1);
30
31   writeln('main before -> i = ', i);
32   writeln('main before -> j = ', j:0:1);
33
34   pass_val(i, j);
35
36   writeln('main -> i = ', i);
37   writeln('main -> j = ', j:0:1);
38
39 end.
40
```

Output:

```
main before -> x = 3
main before -> y = 18.0
        pass_val before -> l = 3
        pass_val before -> a = 18.0
        pass_val -> l = 6
        pass_val -> a = 20.0
main -> x = 3
main -> y = 20.0
main before -> i = 100
main before -> j = 91.1
        pass_val before -> l = 100
        pass_val before -> a = 91.1
        pass_val -> l = 200
        pass_val -> a = 93.1
main -> i = 100
main -> j = 93.1
```