Parameter Passing

Introduction

- Formal and actual parameters
- Parameter Passing
- Examples

Formal and Actual parameters

Function:

```
int func1 (int a, char b, float& c)
 {
        ..
        ..
}
```

Formal Parameters

Function Call:

```
func1(5 * 3, 'A', z);
```

Actual Parameters

Parameter Passing

- Mechanism which determines what kind of association is in between the formal and actual parameters.
- Techniques used for parameter passing are:
  - Pass by value
  - Pass by value-result
  - Pass by reference
  - Pass by name

Parameter Passing

- By value: formal is bound to value of actual (C, Java)
- By reference: formal is bound to location of actual (Pascal, C++ (& parameters)
- By value-result (copy-return): formal is bound to value of actual; upon return from routine, actual gets copy of formal (in-out parameter mode in Ada)
- By name: formal is bound to expression of actual; expression evaluated whenever needed; writes to parameter are allowed (and can affect other parameters!) (Algol)

Example 1:

```
program example ;
var
        global : integer := 10;
        another : integer := 2 ;
procedure confuse (var first , second : integer ) ;
begin
    first := first + global ;
    second := first * global ;
end ;
begin
confuse (global , another ) ;
```

Pass by value: global:=10; another:=2
Pass by value-result: global:=20 ;another:=200
Pass by reference: global:=20 ; another:=400

Example 2:

```
begin
integer n;
procedure p(k: integer);
    begin
    n := n+1;
    k := k+4;
    print(n);
    end;
n := 0;
p(n);
print(n);
end;
```

Pass by value: 1 1
Pass by value-result: 1 4
Pass by reference: 5 5

Example 3

```
begin
integer n;
procedure p(k: integer);
    begin
    print(k);
    n := n+10;
    print(k);
    n := n+5;
    print(k);
    end;
n := 0;
p(n+1);
end;
```

Pass by value: 1 1 1
Pass by name: 1 11 16

Example 4:

```
begin
array a[1..10] of integer;
integer n;
procedure p(b: integer);
    begin
    print(b);
     n := n+1;
    print(b);
    b := b+5;
    end;
a[1] := 10;
a[2] := 20;
a[3] := 30;
a[4] := 40;
n := 1;
p(a[n+2]);
new_line;
print(a);
end;
```

Pass by reference: 30 30 10 20 35 40
Pass by name: 30 40 10 20 30 45

Example 5:

```
int i = 1;
void p(int f, int g)
{
        g++;
        f = 5 * i;
}
int main() {
    int a[] = {0, 1, 2};
    p(a[i], i);
    printf("%d %d %d %d\n", i, a[0], a[1], a[2]);
}
```

Pass by Value: 1 0 1 2
Pass by reference: 2 0 10 2
Pass by name: 2 0 1 10

Example 6:

```
program main;
    i: integer;
    a: array[1..2] of integer;

    procedure f(x:integer, y:integer)
        begin x := x + 1;
        y := y + 1;
        end
begin (* main *)
    i := 1;
    a[1] := 1;
    a[2] := 2;
    f(i,a[i]);
    print(a[1],a[2]);
end;
```

Pass by reference : 2 2
Pass by name : 1 3

Example 7:

```
program one;
    a: array[1..5] of integer;
    i: integer;
    procedure f(x: integer; y: integer)
    begin
        y := y + 1;
        x := x * 2;
        a[i] := a[i] + 15;
    end
begin (* main program *)
    for j := 1 to 5 loop
            a[j] := j;
    end loop;
    i := 1;
    f(a[i], i);
    for j := 1 to 5 loop
            print(a[j]);
    end loop;
end (* main program *)
```

Pass by value: 16 2 3 4 5
Pass by reference: 2 17 3 4 5
Pass by value-result: 2 2 3 4 5
Pass by name: 1 19 3 4 5