Call-value;Call-by-reference

```
int a = 0 ;
int b = 99 ;
void swap(int x, int y){int temp = x;
                            x = y;
                            y = temp;
                       }

void main(){swap(a,b)}
```

Call-value;Call-by-reference

```
int a = 0 ;
int b = 99 ;

void swap(int *x, int *y){int temp = *x;
                           *x = *y;
                           *y = temp;
                         }

void main(){swap(&a,&b)}
```

Call-value;Call-by-reference

```
function f (x) = {x=x+1;return x; }
var y = 0; print (f(y)+y);
```

Call-by-reference

```
procedure swap (var x: integer, var y ; integer);
  var  z : integer;
  begin
   z := x; x : = y; y := z;
 end;
 swap(a,b)

-  compute the address of a and b and assign
   those addresses to x and y

- swap(1,2) ?
```

Call-by-value-result

```
void swap (int x, int y) { int z;
                             z = x; x = y; y=z; }
swap(i,A[i])
```

1. Compute the r-values of i and A[i]. Bind the r-values to the formals. Compute the l-values (locations) of i and A[i]. Save the l-values.

```
x = i; y=A[i]; px = &i; py = &A[i];
```

2. Execute the body

```
   z = x; x = y; y = z;
```

3. The values of the formals are copied back to the l-values saved before

```
   *px=x; *py=y;
```

ALIASING

```
int i=10;
foo(int x, int y){i=y;}
main(){ int A[20];
        i=2;
        A[i]=99;
        foo(i,A[i]);}
```

```
x = i;           x = &i;
y = A[i];        y = &A[i]
px = &i;         i = *y;
py = &A[i];
i = y;
*px = x;
*py = y;
```

Call-by-reference will change the value of i.

Call-by-name:Variable capture problem

```
let
val i =   7
fun P x = let
               val i = 8
            in
                x + i
          end
in
P i
end
```

Call-by-name:Variable capture problem

```
val n = 9

fun f x = x + n

let val n = 8
   in
 f n
end
```