Question 1:

```
procedure f(var A : array of integer; L, R: Integer);
Var
        I, J: Integer;
        P, T: Integer;
begin
        repeat
                I := L;
                J := R;
                P := A[(L + R) shr 1];
                repeat
                        while A[I] < P do
                                Inc(I);
                        while A[J] > P do
                                Dec(J);
                        if I <= J then
                        Begin
                                T := A[I];
                                A[I] := A[J];
                                A[J] := T;
                                Inc(I);
                                Dec(J);
                        End;
                until I > J;
                if L < J then
                        f(A, L, J);
                        L := I;
        until I >= R;
end;
```

Answer to the question 1:

This function takes one integer array and two integer variables as parameters. Those two parameters indicate the starting and ending index of the current sub array. This function sorts the given integer array with O(n log n) complexity. It is partially similar to the quicksort algorithm.

It takes the middle element of the current sub array as pivot. Increase the left index until the left index's element is smaller than the pivot and decrease the right index until the right index's element is bigger than the pivot.

Then swap the bigger one into the right and smaller one into the left side of the pivot.

If the earlier left index value is smaller than current right index value then start making recursive calls with earlier left index and current right index.

Question 2:

```
procedure f( var a : integer; var b : integer );
Begin
  a := a xor b;
  b := a xor b;
  a := a xor b;
end;
```

Answer to the question 2:

This function takes two integer parameters. It's basically a swapping function of two variables without taking help from a temporary variable.