```
Input: matrix A, array b
Output: array x, matrix L, matrix U, matrix P

begin partial_lu(matrix A, array b)
        int n <- A.size()
        matrix u <- zeros_matrix(n,n)
        matrix l <- identity_matrix(n)
        matrix p <- identity_matrix(n)

        for (int k from 0 until n)
                A, p <- search_bigger_and_swap(A, n, k, p)

                for (int i form k + 1 until n)
                        float mult<- A[i][k] / A[k][k]
                        l[i][k] <- mult

                        for (int j from k until n)
                                A[i][j] <- A[i][j] - mult * A[k][j]
                        end for
                end for

                for (int i from 0 until n)
                        u[k][i] <- A[k][i]
                end for
        end for

        matrix pb <- matmul(p, b) // matmul is a function that calculate
                                     the product between matrix
        array z <- solution(l, pb)      // solution is a function that solve
                                     systems of equations
        array x <- solution(u, z)

        return x
end partial_lu

begin search_bigger_and_swap(matrix Ab, int n, int i, matrix p)
        int row = i

    for (int j from i + 1 until n)
        if (absolute_value(Ab[row][i]) < absolute_value(Ab[j][i]))
            row <- j
                end if
    end for

    array temp <- Ab[i]
    array aux <- p[i]
```

```
    Ab[i] <- Ab[row]
    p[i] <- p[row]
    Ab[row] <- temp
    p[row] <- aux

    return Ab, p
end search_bigger_and_swap
```