1. We let \( \minSpot \) be the place at which the minimum remaining element is found. After we find it on the \( i \text{th} \) pass, we just have to interchange the elements in location \( \minSpot \) and location \( i \).

```java
function SelectionSort (a: array of \( n \) integers)
for \( i = 0 \) to \( n-1 \)
{
    \( \minSpot = i \)
    for \( j = i+1 \) to \( n \)
        if \( a[j] < a[\minSpot] \) then \( \minSpot = j \)
    //Interchange the terms
    temp = a[\minSpot]
    a[\minSpot] = a[i]
    a[i] = temp
}
//the array is now in order
```

2. Since \( x^2 \leq x^4 \ \forall \ x > 1 \), we know that \( x^2 \) is \( O(x^4) \) with \( C = 1 \) and \( k = 1 \). Since \( x^2 \) and \( x^4 \) are positive for \( x > 1 \), the absolute signs of the formal definition can be omitted.

On the other hand, if \( x^3 \leq Cx^2 \), then dividing by \( x^2 \), we have \( x^3 \leq C \ \forall \ x > k \) (we can do this simplification by division because \( x^2 > 0 \) for all \( x > k \), where \( k \) is positive). But since this latter condition cannot hold for arbitrarily large \( x \), no matter what the value of the constant \( C \), we conclude that \( x^3 \) is not \( O(x^2) \).

**Remark:** In the formal definition, the constants \( C \) and \( k \) need not to be positive. But remember that if such a pair exists, it is not unique and we can always find some constants that are indeed positive and as large as we like. Hence above we have assumed that \( C \) and \( k \) are both positive if exist.

3. We should find a pair \((C, k)\) such that the following inequality holds:
   \[ |4^x + 65| \leq C \cdot |2^x| \text{ whenever } x > k \]
   Note that \( 4^x + 65 \) and \( 2^x \) are always positive so that absolute signs can be omitted.

   Since \( 2^x \) is always positive, we can divide both side to \( 2^x \). Then, we have \( 2^x + 65/2^x \leq C \). Note that \( 2^x + 65/2^x \) is an unbounded function (since as \( x \) goes to infinity, \( 2^x \) goes to infinity and \( 65/2^x \) to \( 0 \)). Thus when \( x \) is arbitrarily large, we cannot set a constant \( C \) that satisfies the inequality above. Therefore, \( 2^x + 65/2^x \) is not \( O(2^x) \).

4. \( f(x) \) is \( O(5x^2) \) means that \( \exists k_1, C_1 \) positive constants such that \( |f(x)| \leq C_1 \cdot 5x^2, \ \forall \ x > k_1 \). Note that \( 5x^2 \) is always non-negative so we don’t need absolute signs.

   We are required to show that \( f(x) \) is also \( O(4x^4) \). So, we need to show that \( \exists C_2, k_2 \) positive constants such that \( |f(x)| \leq C_2 \cdot 4x^4, \ \forall \ x > k_2 \). Note again that \( 4x^4 \) is always non-negative.

   We can write \( x^2 < x^4, \ \forall \ x > 1 \). Therefore, with \( C_2 = (5/4)C_1 \) and \( k_2 = \max(1, k_1) \) we have the following inequality:
   \[ |f(x)| \leq C_1 \cdot 5x^2 \leq C_2 \cdot 4x^4, \ \forall \ x > k_2 \.
   Thus, \( f(x) \) is indeed \( O(4x^4) \).
5. We need to pick the most rapidly growing term in each sum and discard the rest (including multiplicative factors) in this problem.

In the first sum, $3 \cdot 5^n$ grows more rapidly than $n2^n$ (Note that $5^n$ can be written as $2^n \cdot (5/2)^n$)

For the second sum, first recall that $\log n! \leq n \log n \quad \forall n > 0$

So log $n!$ term grows much slower than $5^n$.

Hence the function is $O(5^n \cdot 5^n)$, more simply $O(25^n)$.

6. To show that $f(x)$ is $O(x)$, we need to find a pair $(C, k)$ such that $|x| \leq C |x|, \forall x > k$.

For $x > 1$, the absolute signs of the formal definition can be omitted. We know that $x \log x < x^2$ and since $x$ is larger than 1, we can write $\sqrt{x \log x} < x$. Therefore, the inequality holds for $C = 5$ and $k = 1$. (We could also find other $C$ and $k$ values that satisfy the inequality).

We then conclude that $f(x)$ is $O(x)$.

How about $5\sqrt{x \log x}$ is $\Omega(x)$? That is equivalent to ask whether $x$ is $O(5\sqrt{x \log x})$. Hence we need to find a pair $(C, k)$ such that $x \log x \leq C |x|, \forall x > k$. If we consider only positive $C$ and $k$, then we can get rid of the absolute signs, and then dividing both sides by $\sqrt{x \log x}$ (we can do this since $\sqrt{x \log x}$ is non-zero for positive $k$), we have $\frac{x}{\sqrt{x \log x}} \leq 5C$, which is not possible for arbitrarily large $x$ since $x/\sqrt{x \log x}$ goes to infinity, and so does its square root, as $x$ goes to infinity. That means we cannot find such positive $(C, k)$ that satisfies the inequality, hence $f(x)$ is not $\Omega(x)$.

7. Before delving into the answers, let’s first remember the definition of the $\Theta$-notation. The $\Theta$-notation says that $f(x)$ is $\Theta(g(x))$ if and only if $f(x)$ is $O(g(x))$ and $g(x)$ is $O(f(x))$. So, we have to show that a function $f(x)$ is $O(g(x))$ and $g(x)$ is $O(f(x))$ to show that $f(x)$ is $\Theta(g(x))$.

$\forall x > 1, \quad 0 \leq 2x^3 + x - 7 \leq 2x^3 + x^3 - 7 \leq 3x^3$

So choosing $C = 3$ and $k = 1$, we can say $2x^3 + x - 7$ is $O(x^3)$.

With $C = 1$ and $k = 3$, $0 \leq x^3 \leq (2x^3 + x - 7) \quad \forall x > k$.

So, $x^3$ is $O(2x^3 + x - 7)$.

Thus, $2x^3 + x - 7$ is $\Theta(x^3)$. $\log_2(x) = \log_2(x) / \log_2(b)$

8. $0 \leq \log_2 x = \log_{10} x / \log_{10} 2 \leq (1/\log_{10} 2) \cdot \log_{10} x \quad \forall x > 1$.

So with $C = 1/\log_{10} 2$ and $k = 1$, $\log_2 x$ is $O(\log_{10} x)$.

$0 \leq \log_{10} x = \log_2 x / \log_2 10 \leq (1/ \log_2 10) \cdot \log_2 x \quad \forall x > 1$.

So with $C = 1/ \log_2 10$ and $k = 1$, $\log_{10} x$ is $O(\log_2 x)$.

Hence, $\log_2 x$ is $\Theta(\log_{10} x)$. 
9.
No. To show this, a counter-example is sufficient.

Suppose that \( f(x) = 2x \) and \( g(x) = x \).
Then clearly \( f(x) \) is \( O(g(x)) \). Now let’s check whether or not \( 3^{2x} \) is \( O(3^x) \).

We must be able to find a pair \((C, k)\) such that the following inequality holds:
\[
|3^{2x}| \leq C \cdot |3^x| \quad \forall x > k.
\]
Note that \( 3^x \) and \( 3^{2x} \) are always positive.
Thus we have (eliminating absolute signs)
\[
3^{2x} \leq C \cdot 3^x \quad \forall x > k
\]
\[
\Rightarrow 3^x 3^x \leq C \cdot 3^x \quad \forall x > k
\]
\[
\Rightarrow 3^x \leq C \quad \forall x > k
\]
Note that it is not possible to find a pair \((C, k)\) that satisfies the above inequality since \( 3^x \) can be made arbitrarily large by increasing \( x \). Hence \( 3^{2x} \) is not \( O(3^x) \).

10. The following uses the formal definition of big-Theta notation:

\( f(x) \) is \( \Theta(g(x)) \) means \( \exists C_1, C_2, k_1 \) positive constants such that
\[
C_1 |g(x)| \leq |f(x)| \leq C_2 |g(x)|, \quad \forall x > k_1.
\]

\( g(x) \) is \( \Theta(h(x)) \) means \( \exists C_3, C_4, k_2 \) positive constants such that
\[
C_3 |h(x)| \leq |g(x)| \leq C_4 |h(x)|, \quad \forall x > k_2.
\]

From these two definitions, we have:
\[
C_1 C_3 |h(x)| \leq C_1 |g(x)| \leq |f(x)| \leq C_2 |g(x)| \leq C_2 C_4 |h(x)|, \quad \forall x > \max(k_1, k_2).
\]
So, with \( C = C_1 C_3, \quad C' = C_2 C_4, \quad \) and \( k = \max(k_1, k_2), \) we have
\[
C |h(x)| \leq |f(x)| \leq C' |h(x)|, \quad \forall x > k.
\]
Thus, \( f(x) \) is \( \Theta(h(x)) \).

Remark: You could also solve this problem by using the definition of big-Theta as given in lecture notes and in the previous exercises 7 and 8, that is, by using the definition of big-O notation.

11. It takes \( n-1 \) comparisons to find the minimum element in the whole array, then \( n-2 \) comparisons to find the minimum among the remaining elements, and so on. Thus the total number of comparisons is \( (n-1) + (n-2) + \ldots + 2 + 1 = n(n-1)/2 \), which is \( O(n^2) \), and which is also \( \Theta(n^2) \). Note that our pseudocode does not involve any arithmetic operations although you might need to include some while implementing the for loops. However that would not change the complexity of the algorithm in terms of any notation.