Question 1. Write an algorithm beautiful(A, n)

Input : An integer array with n elements
such that the best case running time is equal to the worst case running time. Write the algorithm and give your analysis to justify your claim.

Algorithm beautiful(A, n)
  Input Array A of n integers
  Output sum elements of A

```
  sum <- 0
  for i <- 0 to n - 1 do
    sum <- sum + A[i]

  return sum
```

Explanation: In this case, with any input, the best case and the worst case have also the same elements, then the running time is also the same.

Question 2. Order them based on their complexity.

2^n , 2^{2n}, 2^{n+1}, $2^{2^\wedge(n)}$ (Note: ^ stands for exponent operation. Example: 2^n = 2n)

Answer:

- 2^n , 2^{2n}, 2^{n+1} have also the same complexity: $O(2^n)$
- $2^{2^\wedge(n)}$ has the complexity greater than 2^n : $O(2^{2^\wedge(n)})$

Question 3. Mention one algorithm you know for each of the time complexities listed.

$O(1)$, $O(\log n)$, $O(n)$, $O(n \log n)$, $O(n^2)$, $O(n^3)$, $O(2^n)$:

- O(1): assign value from an array index, queue shift, pop
- O(log n): Binary search
- O(n): Linear search

- $O(n \log n)$: Quick sort
- $O(n^2)$: Bubble sort
- $O(n^3)$: No idea. May be used to find the routes between 2 addresses.
- $O(2^n)$: Chess, Chinese chess.

Question 4. Apply Master Theorem and determine the time complexity of

(a) fib(n) shown in slide 48.
(b) binSearch shown in slide 41

Question 5. Practice Master theorem.

It is a very important result in Analysis of algorithms. There are many resources on the internet. Show three different examples covering three possible cases. Show your detailed work.