**1** What is the largest-possible number of inversions a 6-element array can have?

1. 15
2. 21
3. 36
4. 64

(1) is correct. The inversions are the largest when the array is arranged in descending order such that if $x_i < x_j \implies A[x_i] > A[x_j]$. We construct an example [6, 5, 4, 3, 2, 1]. We see that the number of inversions are $5+4+3+2+1 = \sum_{i=1}^{5} i = 15$. In general the largest-possible number of inversions for an array of length $n$ is $\sum_{i=1}^{n-1} i = \frac{(n-1)\cdot n}{2}$.

**2** Suppose the input array $A$ has no split inversions. What is the relationship between the sorted subarrays $C$ and $D$?

1. $C$ has the smallest element of $A$, $D$ the second-smallest, $C$ the third smallest, and so on.
2. All elements of $C$ are less than all elements of $D$.
3. All elements of $C$ are greater than all elements of $D$.
4. There is not enough information to answer this question.

**3** What is the asymptotic running time of the straightforward algorithm for matrix multiplication, as a function of the matrix dimension $n$? Assume that the addition or multiplication of two matrix entries is a constant-time operation.

1. $\Theta(n \cdot logn)$
2. $\Theta(n^2)$
3. $\Theta(n^3)$
4. $\Theta(n^4)$

**4** Suppose that we correctly implement the $ClosetSplitPair$ subroutine in $O(n)$ time. What will be the overall running time of $ClosetPair$ algorithm?

1. $O(n)$
2. $O(n \cdot logn)$
3. $O(n \cdot (logn)^2)$
4. $O(n^2)$

`FastPower`

**Input:** positive integers $a$ and $b$.
**Output:** $a^b$.

---

**if** $b = 1$ **then**
    return $a$
**else**
    $c := b \cdot b$
    $ans :=$ `FastPower`$(c, \lfloor b/2 \rfloor)$
**if** $b$ is odd **then**
    return $a \cdot ans$
**else**
    return $ans$

---

Figure 0.1: FastPower

**5** Consider the following pseduocode for calculating $a^b$ where $a$ and $b$ are positive integers.

Assume for this problem that each multiplication and division can be performed in constant time. What is the asymptotic running time of this algorithm as a function of $b$?

1. $\Theta(logb)$
2. $\Theta(\sqrt{b})$
3. $\Theta(b)$
4. $\Theta(b \cdot logb)$

**6** You are given a *unimodal* array of $n$ distinct elements, meaning that its entries are in increasing order until its maximum element, after which its elements are in decreasingorder. Give an algorithm to compute the maximum element of a unimodal array that runs in $O(logn)$ time.

**7** You are given a sorted array $A$ of $n$ distinct integers which can be positive, negative, or zero. You want to decide whether or not there is an index $i$ such that $A[i] = i$. Design the fast algorithm you can for solving this problem

**8** You are given an $n-by-n$ grid of distinct nmbers. A number is a *local minimum* if its is smaller than all its neighbors. Use the divide-and-conquer algorithm to compute a local minimum with only $O(n)$ comparison between pairs of numbers