CIT 596: ALGORITHMS & COMPUTATION

An Iterative Algorithm: Insertion Sort

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs

Sorting an Array

Input: an array A of n numbers, indexed from 1 to n

Output: A permutation (reordering) of A's entries such that

Assignment Project Exam Help

$$A[1] \le A[2] \le \ \ldots \ \le A[n]$$

https://tutorcs.com

Example: Given the input array

WeChat: cstutorcs

16	7	13	22	4	11	25	8	5	17

the output should be

4	5	7	8	11	13	16	17	22	25

Insertion Sort

Starting from the left, put each entry in its correct position among the entries to its left.

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs

16	7	13	22	4	11	25	8	5	17

Describing the Algorithm

An example is not a description

When you fully understand an algorithm, you should be able to describe it to both computers and humans.

Assignment Project Exam Help

- Implementation in code has obvious practical value, and it can be a useful exercise to address every detail and resolved every ambiguity.

https://tutorcs.com

- Our focus in this class is on describing algorithms to humans, which is more difficult in some ways! You'll need to write clearly and emphasize high-level ideas without overlooking important details. Two main options:

WeChat: cstutorcs

  1. Describe your algorithm in paragraph form. As we get further into the course, this will be the preferred option.
  2. Write pseudocode, which is structured like code but intended for humans to read.

Pseudocode

INSERTIONSORT(A)
    for $i = 1$ to $length(A)$
        $j = i$
        while $j > 0$ and $A[j-1] > A[j]$
            swap $A[j]$ and $A[j-1]$
            $j = j - 1$

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

No specific stylistic rules. The goals are readability and clarity.

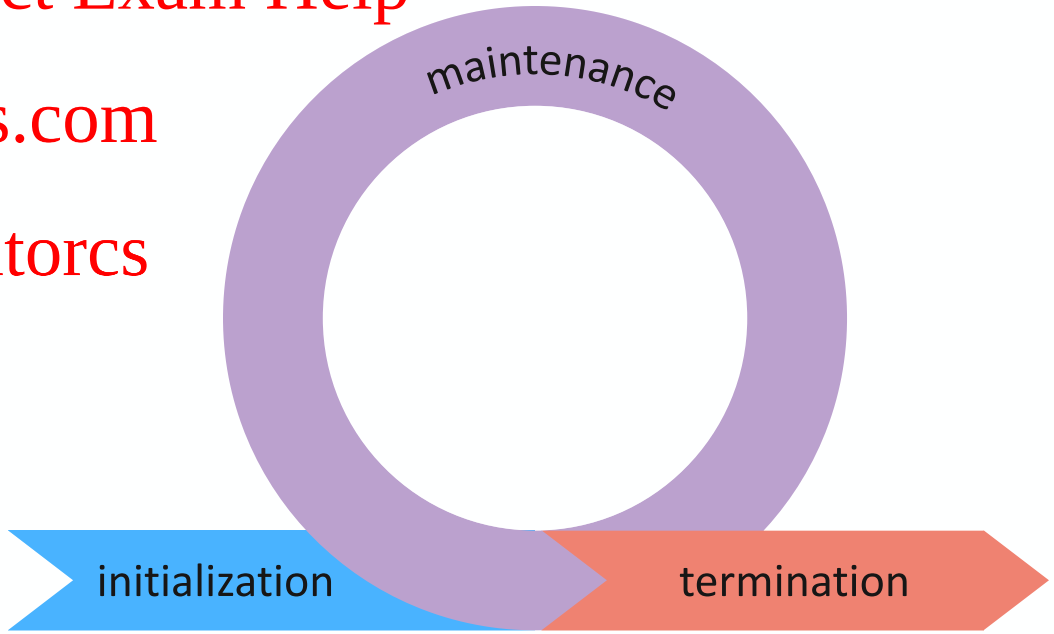
- Detailed enough that any competent programmer could quickly implement it in a language of their choice.
- Avoids language-specific idioms or symbols.

A Proof Technique for Iterative Algorithms

Loop invariant: an induction hypothesis carefully expressing the notion that "the algorithm is currently on track for a correct solution."

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

- **Initialization:** The invariant holds for the first iteration of the loop.
- **Maintenance:** If the invariant holds for a non-final iteration of the loop, then it also holds for the next iteration.
- **Termination:** If the invariant holds for the final iteration of the loop, then the algorithm is correct.



Proof of Correctness for Insertion Sort

Loop invariant: *After the i^{th} iteration of the* **for** *loop, the subarray $A[1, \ldots, i]$ contains the original entries of that subarray in sorted order*

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs

- **Initialization:** After the first iteration of the **for** loop, $i = 1$. No swaps have occurred, so $A[1]$ still has its original value, and a subarray of length 1 is trivially in sorted order.

Proof of Correctness for Insertion Sort

- **Maintenance:** Suppose the invariant holds for the i^{th} iteration of the **for** loop, for some $1 \leq i < n$. Let x be the number in $A[i+1]$ at the beginning of the $(i+1)^{\text{st}}$ iteration. The **while** loop pushes x to the left until either $x = A[1]$ or $x = A[j] \leq A[j-1]$. At that point, $A[1, \ldots, i+1]$ contains its original entries in sorted order.

Assignment Project Exam Help
https://tutorcs.com
WeChat: cstutorcs

- **Termination:** After the last iteration of the **for** loop, $i = n$ so the loop invariant states that $A = A[1, \ldots, n]$ contains its original entries in sorted order, which is the correct output condition for an array-sorting algorithm.