|  |  |
| --- | --- |
| **Application/ Program name:** |  |
| **Written by:** |  |

|  |
| --- |
| **Purpose or problem definition:** |
|  |
|  |
| **Program Procedures:** |
|  |
|  |
| **Algorithm/Processing/Conditions:** |
| **Inputs:** |
|  |
| **Processes:** |
|  |
| **Outputs:** |
|  |
|  |
| **Notes & Restriction:** |
| Linked List  From Wikipedia <https://en.wikipedia.org/wiki/Linked_list>  “In [computer science](https://en.wikipedia.org/wiki/Computer_science), a **linked list** is a linear collection of data elements whose order is not given by their physical placement in memory. Instead, each element [points](https://en.wikipedia.org/wiki/Pointer_(computer_programming)) to the next. It is a [data structure](https://en.wikipedia.org/wiki/Data_structure) consisting of a collection of [nodes](https://en.wikipedia.org/wiki/Node_(computer_science)) which together represent a [sequence](https://en.wikipedia.org/wiki/Sequence). In its most basic form, each node contains: [data](https://en.wikipedia.org/wiki/Data_(computing)), and a [reference](https://en.wikipedia.org/wiki/Reference_(computer_science)) (in other words, a *link*) to the next node in the sequence. This structure allows for efficient insertion or removal of elements from any position in the sequence during iteration. More complex variants add additional links, allowing more efficient insertion or removal of nodes at arbitrary positions. A drawback of linked lists is that access time is linear (and difficult to [pipeline](https://en.wikipedia.org/wiki/Instruction_pipelining)). Faster access, such as random access, is not feasible. [Arrays](https://en.wikipedia.org/wiki/Array_data_structure) have better [cache locality](https://en.wikipedia.org/wiki/Locality_of_reference) compared to linked lists.” |
|  |
| **Comments:** |
|  |