|  |  |  |
| --- | --- | --- |
|  | **STRING BUFFER** | **STRING BUILDER** |
| 1 | Introduced in initial release of Java | Introduced in Java 5 |
| 2 | Synchronized | Non - Synchronized |
| 3 | Thread safe and slow | non thread safe and fast |
| 4 | Mutable (Modify string without creating an object) | Mutable |
| 5 | Stored in Heap Memory | Stored in Heap Memory |
|  |  |  |
|  |  |  |
|  |  |  |
|  | **STACK MEMORY** | **HEAP MEMORY** |
| 1 | Used to store item which have very short life like local variable, reference variable of object | Used to allocate to store object and JRE class |
| 2 | Follow LIFO order | Dynamic allocation there is no fixed pattern for allocating and deallocating blocks in memory |
| 3 | Increase stack memory size by using JVM parameter XSS | We can increase or decrease heap memory size by using JVM options XMS and XMX |
| 4 | Variables are visible to only to owner thread | visible to all thread |
| 5 | JVM will throw java.lang.StackOverFlowError | JVM will Throw java.lang.OutOfMemoryError |
|  |  |  |
|  |  |  |
|  | **VIRTUAL MEMORY** | **CACHE MEMORY** |
| 1 | Increase Main memory capacity | Increase CPU access speed |
| 2 | It is a technique and involves hard disk and it slower to access | It is a memory unit and is very fast to access |
| 3 | OS manages virtual memory | CPU and related hardware manage cache memory |
| 4 | Size of virtual memory is larger than cache memory | small in size |
| 5 | keeps the program which are not getting accommodated in main memory | keeps recently used data |
| 6 | mapping structures are required in the virtual memory to map virtual address to physical address | There is no need of mapping structure in the cache memory |
|  |  |  |
|  | **STACK** | **ARRAY** |
| 1 | Type of linear data structure that is represented by a collection of pieces that are arranged in a predetermined sequence | An array us a collection of data values that are associated to be another and termed elements. Each element is recognized by an indexed array |
| 2 | Push, pop and peak are the operations that one can perform on a stack | It has a wide range of techniques or operations that can be applied to it. Such as sorting, traversing, push, pop etc., |
| 3 | The size of a stack is dynamic | The size of an array is fixed |
| 4 | Insertion and deletion takes place only on top | We can serve any point as starting for insertion and deletion |
| 5 | Follows LIFO order | Get the value by the index. Eg.[1] |
| 6 | We can create a stack using an array | Stacks cannot be used to implement an array |
| 7 | Several data types used | Same data types can be used |