**C++ Cheat Sheet**

|  |  |
| --- | --- |
| CHAR\_BIT | 8 |
| SCHAR\_MIN | -128 |
| SCHAR\_MAX | +127 |
| UCHAR\_MAX | 255 |
| CHAR\_MIN | -128 |
| CHAR\_MAX | +127 |
| MB\_LEN\_MAX | 16 |
| SHRT\_MIN | -32768 |
| SHRT\_MAX | +32767 |
| USHRT\_MAX | 65535 **(because no bit reserved for sign)** |
| INT\_MIN | -2147483648 |
| INT\_MAX | +2147483647 |
| UINT\_MAX | 4294967295 **(because no bit reserved for sign)** |
| LONG\_MIN | -9223372036854775808 |
| LONG\_MAX | +9223372036854775807 |
| ULONG\_MAX | 18446744073709551615 **(because no bit reserved for sign)** |

MAX = 1e9+7 or **1000000007**

|  |
| --- |
| **MAX value of N Time complexity** |
| 10^9 O(log N) or Sqrt(N) |
| 10^8 O(N) Border case |
| 10^7 O(N) Might be accepted |
| 10^6 O(N) Perfect |
| 10^5 O(N \* log N) |
| 10^4 O(N ^ 2) |
| 10^2 O(N ^ 3) |
| <= 160 O(N ^ 4) |
| <= 18 O(2N\*N2) |
| <= 10 O(N!), O(2N) |

|  |
| --- |
| **NOTE: Short names include DT (Data Type)** |

**Containers**

1. **Vectors**

|  |  |  |
| --- | --- | --- |
| Member Functions | Return Values | Command Brief |
| vector<DT> v |  | Default, Empty vector |
| vector<DT> v(size) |  | Fill constructor |
| vector<DT> v(size, init\_value) |  | Filled constructor |
| vector<DT> v2(itr1, itr2) |  | Deep copy, range based |
| vector<DT> v1(v2) |  | Deep copy, copy constructor |
| vector<DT>() | Returns Empty vector |  |
| v.assign(size, value) |  | Same as filled constructor but a function |
| v.assign(itr1, itr2) |  | Range based filling |
| vector<DT> v = { \_\_\_\_\_\_\_\_ } |  | Initialization |
| v.begin() | Iterator pointing to first element |  |
| v.end() | Iterator pointing to the post last element |  |
| v.rbegin() | Reverse Iterator pointing to the last element and this iterator traverse in reverse. |  |
| v.rend() | Reverse Iterator pointing to the element preceding first element and this iterator traverse in reverse. |  |
| v.size() | Unsigned Integer |  |
| v.max\_size() |  | Maximum potential size of container |
| v.capacity() | Unsigned Integer | 2^n size |
| v.resize(size, value) |  | Makes and vector to the given size while filling empty spaces with value |
| v.shrink\_to\_fit() |  | Resizes to accommodate available elements. Capacity=Size. |
| v.at(index) | DT value | Returns value at index |
| Iterator loop, \*itr is the element | \*itr=value and itr is address location |  |
| v[index] |  |  |
| v.data() | Pointer to vector’s start, pointing to first element |  |
| v.front() | Returns leftmost element |  |
| v.back() | Returns rightmost element |  |
| v.clear() | Void | Removes all elements in the container. Size = 0. Capacity is intact. |
| v.emplace(itr, value) | Returns iterator to newly inserted element. | Inserts value before the location pointed by itr. |
| v.emplace\_back(value) | Void | Similar to push\_back(value) |
| v.empty() | Boolean | If container is empty or not |
| v.erase(itr1, itr2) | Returns iterator pointing to element after the one which was erased | If one argument is given, one element will be erased at location pointed by it. In case of 2, range is erased as [itr1, itr2). |
| Member Functions | **Return Values** | **Command Brief** |
| v.insert(itr, value) | Returns random access iterator pointing to first of newly inserted element(s). | Pushes before itr with value |
| v.insert(itr, count, value) | Returns random access iterator pointing to first of newly inserted element(s). | Pushes before itr with count number of values |
| v.insert(itr, itr\_1, itr\_2) | Returns random access iterator pointing to first of newly inserted element(s). | Pushes before itr with ellements between [itr\_1, itr\_2) |
| v.push\_back(value) | Void | Appends value on right |
| v.pop\_back() | Void | Pops value from right, size--; |