|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | | java |   1).Java does not support pointers, templates, unions, operator overloading, structures etc.  The Java language promoters initially said "No pointers!", but when many programmers questioned how you can work without pointers, the promoters began saying "Restricted pointers." Java supports what it calls "references". References act a lot like pointers in C/C++ languages but you cannot perform arithmetic on pointers in Java. References have types, and they're type-safe. These references cannot be interpreted as raw address and unsafe conversion is not allowed. | |  | | --- | | C++ |   1).C++ supports structures, unions, templates, operator overloading, pointers and pointer arithmetic. |
| 2).Java support automatic garbage collection. It does not support destructors as C++ does. | 2).C++ support destructors, which is automatically invoked when the object is destroyed. |
| 3).Java does not support conditional compilation and inclusion. | 3).Conditional inclusion (#ifdef #ifndef type) is one of the main features of C/C++. |
| 4).Java has built in support for threads. In Java, there is a threat class that you inherit to create a new thread and override the run() method. | 4).C++ has no built in support for threads. C++ relies on non-standard third-party libraries for thread support. |
| 5).Java does not support default arguments. There is no scope~resolution~operator (::) in Java. The method definitions must always occur within a class, so there is no need for scope resolution there either. | 5).C++ supports default arguments. C++ has scope~resolution~operator (::) which is used to to define a method outside a class and to access a global variable within from the scope where a local variable also exists with the same name. |
| 6).There is no *goto* statement in Java. The keywords const and goto are reserved, even though they are not used. | 6).C++ has *goto* statement. However, it is not considered good practice to use of*goto* statement. |
| 7).Java doesn't provide multiple inheritance, at least not in the same sense that C++ does. | 7).C++ does support multiple inheritance. The keyword virtual is used to resolve ambiguities during multiple inheritance if there is any. |
| 8).Exception handling in Java is different because there are no destructors. Also, in Java, try/catch must be defined if the function declares that it may throw an exception. | 8).While in C++, you may not include the try/catch even if the function throws an exception. |