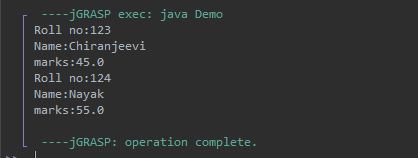
|  |  |
| --- | --- |
| **Interpreter** | **Compiler** |
| Interpreter translates just one statement of the program at a time into machine code. | Compiler scans the entire program and translates the whole of it into machine code at once. |
| An interpreter takes very less time to analyze the source code. | A compiler takes a lot of time to analyze the source code. |
| An interpreter does not generate an intermediary code. Hence, an interpreter is highly efficient in terms of its memory. | A compiler always generates an intermediary object code. It will need further linking. Hence more memory is needed. |
| Keeps translating the program continuously till the first error is confronted. If any error is spotted, it stops working and hence debugging becomes easy. | A compiler generates the error message only after it scans the complete program and hence debugging is relatively harder while working with a compiler. |
| Interpreter are used by programming languages like Ruby and Python. | Compilers are used by programming languages like C and C++. |

1. **Difference Between interpreter and compiler**

**2.**

class student  
{  
 int roll;  
 String name;  
 float marks;  
   
 public void input(int r,String n,float m)  
 {  
 roll=r;  
 name=n;  
 marks=m;  
 }  
 public void display()  
 {  
 System.out.println("Roll no:"+roll);  
 System.out.println("Name:"+name);  
 System.out.println("marks:"+marks);  
 }  
}  
class Demo  
{  
 public static void main(String args[])  
 {  
 student ob=new student();  
 student ob1=new student();  
 ob.input(123,"Chiranjeevi",45);  
 ob.display();  
 ob1.input(124,"Nayak",55);  
 ob1.display();  
 }  
}

**Output**

****