|  |  |
| --- | --- |
| **Procedural Oriented Programming** | **Object-Oriented Programming** |
| It is based on functions. | It is based on real-world objects. |
| It follows a top-down approach. | It follows a bottom-up approach. |
| It is less secure because there is no proper way to hide data. | It provides more security. |
| Data is visible to the whole program. | It encapsulates the data. |
| Reuse of code is not allowed. | The code can be reused. |
| Modification and extension of code are not easy. | We can easily modify and extend code. |
| Examples of POP are C, VB, FORTRAN, Pascal, etc. | Examples of OOPs are C++, Java, C#, .NET, etc. |

|  |  |
| --- | --- |
| **Object-oriented Programming** | **Structural Programming** |
| It follows a bottom-up approach. | It follows a top-down approach. |
| It provides data hiding. | Data hiding is not allowed. |
| It is used to solve complex problems. | It is used to solve moderate problems. |
| It allows reusability of code that reduces redundancy of code. | Reusability of code is not allowed. |
| It is based on objects rather than functions and procedures. | It provides a logical structure to a program in which the program is divided into functions. |
| It provides more security as it has a data hiding feature. | It provides less security as it does not support the data hiding feature. |
| More abstraction more flexibility. | Less abstraction less flexibility. |
| It focuses on data. | It focuses on the process or logical structure. |