Atualmente existem dois tipos de programação considerados usuais, a estruturada e a orientada a objetos. O primeiro paradigma é centrado em funções e no controle de fluxo com etapas lógicas lineares, enquanto o segundo foca em objetos e classes que interagem entre si através de mensagens, aproximando as estruturas de um programa à estrutura das coisas do mundo real (PROGRAMAE, 2024).

De acordo com dados históricos, os estudos sobre programação orientada a objetos surgiram no final da década de 60 no cenário da “crise do software”. Esse paradigma não possui um único criador, mas sabe-se que a linguagem Simula 67 foi a primeira a introduzir os conceitos de objetos, classes e herança. Esses conceitos foram aprimorados por meio da linguagem Smalltalk, desenvolvida no início da década de 70 pelo Alan Kay, conhecido como um dos pioneiros da POO, no Xerox PARC; mas só ganharam destaque a partir da década de 80 ao serem aplicados nas linguagens Java e C++ (METROPOLE, 2022).

Dentre as vantagens fornecidas por esse paradigma, encontram-se a modelagem do mundo real, que torna o desenvolvimento mais intuitivo e facilita a comunicação interna; a reutilização do código, pois através da herança e do polimorfismo é possível utilizar métodos e classes já existentes em outro contexto; a manutenibilidade simplificada, a possibilidade de desenvolvimento colaborativo, a extensibilidade e a rastreabilidade (BONATINI, 2023).

Em POO, as classes são estruturas que atuam como moldes ao agrupar itens com as mesmas propriedades e funcionalidades e por meio destas criam-se os objetos, que são as instâncias das classes. Os objetos terão os atributos, que correspondem às suas características; os métodos, que representam as ações e os comportamentos que eles poderão realizar e esses apresentam níveis de visibilidade, que determinam o acesso às informações, podendo ser público, privado ou protegido (NORMANDO, 2024).

Neste cenário surgem os seguintes conceitos cruciais, conhecidos como pilares da programação orientada a objetos: abstração, herança, polimorfismo e encapsulamento.

A abstração visa à simplificação ao modelar comportamentos e características concentrando-se apenas nas informações do objeto que são essenciais e relevantes dentro de um contexto. A herança fornece a funcionalidade da reutilização de métodos e atributos a partir de uma hierarquização de classes. O polimorfismo promove a capacidade de um objeto executar diferentes comportamentos a partir do mesmo método em situações distintas. E o encapsulamento visa controlar o acesso aos métodos e atributos da classe através dos níveis de visibilidade, permitindo a ocultação de dados e da estrutura.; por meio desses conceitos, são gerados os benefícios da utilização do paradigma abordado (SILVA, 2023).