As origens do paradigma orientado a objetos:

Entre 1966 e 1968, na Noruega, surgiu a linguagem SIMULA como uma ampliação de sua linguagem antecedente, ALGOL. SIMULA nasceu para facilitar a criação de sistemas modelados através da junção de várias partes produzidas distintamente, ainda assim ela não possuía alguns dos principais conceitos das linguagens orientadas a objetos como polimorfismo e herança. Na década de 70 as pesquisas do projeto Dynabook em um centro de pesquisa da XEROX na Califórnia resultam na primeira versão da linguagem que posteriormente ficaria conhecida como a pioneira em orientação a objetos, a Smalltalk (1972 primeira versão).

Os conceitos derivados da linguagem SIMULA como classes e a divisão em partes menores se juntavam a base advinda da linguagem LISP para fazer da Smalltalk uma linguagem de fácil aprendizado e com uma sintaxe simples para programar. Somente na versão de 76 a linguagem passou a implementar a ideia de herança.

Conceitos principais de POO:

O paradigma orientado a objetos parte da representação do mundo real como um conjunto de componentes/objetos/coisas/entidades que interagem entre si e não apenas a junção de vários processos estruturados.

Com o uso dos conceitos: Abstração, encapsulamento, modularização, relacionamento, hierarquização, polimorfia, a programação orientada a objetos busca representar da forma mais fiel os acontecimentos do mundo real através de suas classes.

Classes em POO são “moldes” para objetos importantes da realidade que mereçam uma representação no ambiente observado. Portanto, um objeto é uma instância de alguma classe.

Classes:

As classes são uma generalização de objetos com características comuns entre si. Elas são constituídas por dois conjuntos, um de atributos e outro de métodos.

Atributos: São as características de um objeto. Os atributos armazenam valores como variáveis que determinam o estado do objeto. Atributos são inacessíveis e a única maneira de modificá-los é através dos métodos da classe.

Métodos: Constituem uma interface de ações executadas pelo objeto. Um método é uma chamada que engatilha um estímulo no objeto e até pode gerar uma mudança no mesmo.

Por irônico que parece, o método fundamental para paradigma orientado a objetos se dá na criação de classes e não propriamente dos objetos. A principal vantagem das classes é poder reutilizá-las mais de uma vez em diferentes códigos com alterações quando necessárias. Outra propriedade das classes é a hierarquia que pode gerar subclasses e superclasses.

Superclasse e subclasse:

Classes podem compartilhar características suas para outras classes filhas que tenham mais especializações. Uma superclasse é o pai de alguma classe que se derivou desta, enquanto essa classe filha que herdou as características da classe maior é chamada de subclasse.

Conceitos para classes:

Abstração: Desempenha um papel importante na criação das classes porque traz o princípio de que nem toda característica de uma classe é importante para o sistema. Cabe ao programador filtrar as informações e identificar o que será necessário e usado para representar o objeto real daquela classe a ser criada.

Encapsulamento: Um grande avanço em relação a programação estruturada é a possibilidade de permitir o objeto disponível e funcional sem a necessidade de acessar seu interior. Apenas os métodos de uma classe são capazes de modificar o objeto dela e ainda assim objetos diferentes podem interagir entre si, bastando apenas que um deles tenha acesso a interface de métodos do outro.

Herança: Como explicado antes, classes podem possuir subclasses que possuam características mais específicas para o objeto, as classes filhas herdam as características de sua(s) superclasse(s), não sendo necessário repetir o que já está dentro das classes ancestrais. Dentro de herança uma classe pode possuir um único ancestral, herança simples, ou mesmo mais de um ancestral, herança múltipla, fazendo-a possuir as características das duas superclasses.

Polimorfismo: Um mesmo método pode ser usado em diferentes classes de forma específica e diferente para cada objeto.

Programação orientada a objetos X Programação estruturada:

A programação estruturada revolucionou em sua época com a ideia central de dividir um grande problema em partes menores e por meio de conceitos como sequência, decisão e iteração possibilitar a criação de softwares mais robustos com um custo de tempo e do projeto mais vantajoso. Ainda assim, a falta de flexibilidade e dificuldade em determinar a prioridade entre estruturas ou processos diminui o desempenho desse paradigma em grandes projetos.

A programação orientada a objetos foca em identificar os principais componentes que um sistema possui e suas interações, a mesma metodologia usada para definir a lógica do sistema também pode ser usada na implementação sem a necessidade do analista adaptar a programação e estrutura dos dados como em paradigma estruturado.

Vale ressaltar que a programação orientada a objetos pode diversas vezes ser uma desvantagem em relação ao paradigma estruturado. Por exemplo, para a programação de

um código que realiza um cálculo, dividir o problema em partes menores a serem solucionadas vai ser mais prático do que tentar tornar suas partes em objetos.

Conclusão:

O paradigma orientado a objetos surgiu da necessidade de distribuir uma extensa tarefa entre uma grande equipe, sua primeira versão em linguagem de programação queria trazer uma sintaxe mais compreensível e fácil de ser aprendida por outros programadores. No geral as mesmas características do passado que a originaram a mantêm como um ótimo recurso para programadores da atualidade.

A facilidade em reutilizar partes do código; Modificar classes e realizar a manutenção do código com a certeza de que as modificações são restritas ao objeto do contexto; A facilidade em criar uma harmonia no projeto final programado por uma equipe diversa de programadores, caracterizam o paradigma orientado a objetos.

Bibliografia:

FARINELLI, Fernanda. Conceitos básicos de programação orientada a objetos. 2007. Instituto Federal Sudeste de Minas Gerais. https://scholar.google.com.br/citations?view_op=view_citation&hl=pt-BR&user=b2x6tLwAAAAJ&citation_for_view=b2x6tLwAAAAJ:zYLM7Y9cAGgC

ABDALA, Daniel Duarte. WANGENHEIM, Aldo von. Conhecendo o Smalltalk, versão revisada 1- Aldo. 2001. https://www.researchgate.net/publication/262882317_Conhecendo_o_Smalltalk_-_Todos_os_Detalhes_da_Melhor_Linguagem_de_Programacao_Orientada_a_Objetos

OBERLEITNER, Allen. MASIERO, Andrey Araujo. Programação orientada a objetos. SENAC. https://books.google.com.br/books?hl=pt-BR&lr&id=P9IiEAAAQBAJ&oi=fnd&pg=PT5&dq=programa%C3%A7%C3%A3o+orientada+a+objeto&ots=V2Fwdx6pDn&sig=thlZ00A_x23xxgksXcwJbIzK13s&pli=1#v=onepage&q=programa%C3%A7%C3%A3o%20orientada%20a%20objeto&f=false

JONATHAN, Miguel. Introdução à programação orientada a objetos com Smalltalk.1994. Universidade Federal do Rio de Janeiro. https://dcc.ufrj.br/~jonathan/smalltalk/Introd-a-POO-com-Smalltalk-1994.pdf