Object-Oriented Programming (OOP)

Object-Oriented Programming (OOP) ծրագրավորման պարադիգմ է, որը պտտվում է «օբյեկտների» հայեցակարգի շուրջ, որը կարող է ամփոփել տվյալները և վարքագիծը: Այն ընդգծում է կոդի կազմակերպումը մոդուլային միավորների մեջ, որոնք կոչվում են օբյեկտներ, որոնցից յուրաքանչյուրը ներկայացնում է դասի օրինակ: Այս պարադիգմում ծրագրակազմը կառուցված է այս օբյեկտների տեսանկյունից, որոնք կարող են միմյանց հետ շփվել սահմանված միջերեսների միջոցով:

Օբյեկտ-կողմնորոշված ծրագրավորման (OOP) նկարագրությունը.

1. **Օբյեկտներ**. OOP-ում օբյեկտը դասի (class) օրինակ է: Դասը նախագիծ է, որը սահմանում է օբյեկտի հատկությունները (տվյալների դաշտերը կամ ատրիբուտները) և վարքագիծը (մեթոդներ կամ գործառույթներ): Օբյեկտները պահում են տվյալներ՝ կապված իրենց հատկությունների հետ և կարող են կատարել իրենց մեթոդներով սահմանված գործողություններ:
2. **Դասեր (Classes).** Դասը օգտագործողի կողմից սահմանված տվյալների տեսակ է, որը սահմանում է օբյեկտների կառուցվածքը և վարքագիծը: Այն ծառայում է որպես նախագիծ կամ կաղապար՝ օբյեկտներ ստեղծելու համար: Դասերը ներառում են տվյալներ (հատկանիշներ) և վարքագիծ (մեթոդներ)՝ կապված որոշակի անձի կամ հայեցակարգի հետ:
3. **Աբստրակցիա.** Աբստրակցիան հիմնարար հասկացություն է OOP-ում, որը թույլ է տալիս ծրագրավորողին ներկայացնել օբյեկտի էական հատկանիշները՝ թաքցնելով ավելորդ մանրամասները: Այն կենտրոնանում է այն բանի վրա, թե ինչ է անում օբյեկտը, այլ ոչ թե ինչպես է այն հասնում իր ֆունկցիոնալությանը:
4. **Ժառանգություն (Inheritance).** Ժառանգությունը մեխանիզմ է OOP-ում, որը թույլ է տալիս դասին (կոչվում է ստացված կամ ենթադաս) ժառանգել հատկություններ և վարքագիծ մեկ այլ դասից (կոչվում է բազային կամ գերդաս): Սա նպաստում է կոդի վերօգտագործմանը և դասերի միջև հաստատում է «is-a» հարաբերություն:

5. **Պոլիմորֆիզմ.** Պոլիմորֆիզմը նշանակում է «բազմաթիվ ձևեր» և թույլ է տալիս օբյեկտներին վերաբերվել որպես իրենց մայր դասի օրինակներ կամ որպես իրենց հատուկ ենթադասերի օրինակներ: Սա թույլ է տալիս մեկ ինտերֆեյս օգտագործել գործողությունների ընդհանուր դասի համար՝ ապահովելով ճկունության և ընդարձակելիության բարձր մակարդակ:
6. **Էնկապսուլյացիան** տվյալների (ատրիբուտները) և մեթոդների (ֆունկցիաներ) միավորումն է, որոնք գործում են տվյալ տվյալների վրա մեկ միավորի (դասերի) շրջանակներում: Այն օգնում է պաշտպանել տվյալները և վերահսկել դրանց հասանելիությունը՝ տրամադրելով մուտքի փոփոխիչներ, ինչպիսիք են հանրային, մասնավոր և պաշտպանված:

Ինչպես է գործում OOP

1. **Դասակառգում.** Օբյեկտ ստեղծելու համար անհրաժեշտ է դասակարգել: Սա ներառում է դասի օրինակի ստեղծում՝ օգտագործելով նոր օպերատորը այնպիսի լեզուներով, ինչպիսին է C++-ը: Հիշողության և սկզբնավորման այս բաշխումը կարգավորում է ծրագրում օգտագործվող օբյեկտը:
2. **Հատկությունների և մեթոդների հասանելիություն.** Երբ օբյեկտը ստեղծվի, դուք կարող եք մուտք գործել նրա հատկությունները (ատրիբուտները) և մեթոդները (ֆունկցիաները)՝ օգտագործելով կետային նշումը կամ սլաքի օպերատորը (->), կախված նրանից, թե օբյեկտը հասանելի է ուղղակիորեն, թե ցուցիչի միջոցով:
3. **Օբյեկտների միջև հաղորդակցություն.** օբյեկտները փոխազդում են միմյանց հետ՝ օգտագործելով մեթոդներ այլ օբյեկտների վրա՝ փոխանցելով տվյալներ որպես պարամետրեր: Այս հաղորդակցությունը հեշտացվում է լավ սահմանված միջերեսների միջոցով և օգնում է հասնել ծրագրի ցանկալի ֆունկցիոնալությանը:

OOP օգտագործումը օրինակներում.

1. **Բանկային հաշվի համակարգ.** բանկային հավելվածում դուք կարող եք մոդելավորել բանկային հաշիվը որպես դաս: Դասը կպարունակի այնպիսի հատկություններ, ինչպիսիք են հաշվի համարը, հաշվի տիրոջ անունը, մնացորդը և մեթոդներ, ինչպիսիք են deposit, withdraw և getBalance:
2. **Խաղի մշակում.** Խաղերի մշակման մեջ OOP-ը լայնորեն օգտագործվում է տարբեր խաղի տարրեր մոդելավորելու համար: Օրինակ, խաղը կարող է ունենալ դասեր խաղացողների, թշնամիների, զենքերի, մակարդակների և այլնի համար: Յուրաքանչյուր դաս կունենա իր դերին հատուկ հատկություններ և մեթոդներ:

3. **Էլեկտրոնային առևտրի համակարգ.** Էլեկտրոնային առևտրի համակարգում OOP-ը կարող է օգտագործվել ապրանքների, հաճախորդների, պատվերների և վճարումների մշակման մոդելավորման համար: Այս սուբյեկտներից յուրաքանչյուրը կարող է ներկայացվել դասի միջոցով՝ համապատասխան հատկություններով և մեթոդներով՝ դրանց ֆունկցիոնալությունը կառավարելու համար:
4. **Աշխատակիցների կառավարման համակարգ.** Աշխատակիցների կառավարման համակարգը կարող է օգտագործել OOP-ը աշխատողների, բաժինների, դերերի և աշխատավարձերի մոդելավորման համար: Աշխատակիցների համար նախատեսված դասերը կարող են պարունակել մանրամասներ, ինչպիսիք են անունը, պաշտոնը, աշխատավարձը և առաջխաղացման և բոնուսների հաշվարկման մեթոդները:

Օբյեկտ-կողմնորոշված ծրագրավորումը խթանում է մոդուլյարությունը, կրկնակի օգտագործման և ընդարձակելիությունը ծրագրային ապահովման մշակման մեջ՝ դարձնելով այն լայնորեն ընդունված հարացույց ժամանակակից ծրագրավորման մեջ: Ստեղծելով դասեր, որոնք վերացական են իրական աշխարհի սուբյեկտները, ծրագրավորողները կարող են արդյունավետորեն կառավարել և կազմակերպել բարդ համակարգեր՝ հանգեցնելով ավելի պահպանվող և մասշտաբային կոդերի բազաների: