ДНІПРОВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ОЛЕСЯ ГОНЧАРА

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ МАТЕМАТИКИ ТА МАТЕМАТИЧНОЇ КІБЕРНЕТИКИ

ЗВІТ ПРО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 2

З дисципліни «Сучасні середовища програмування»

Перший (бакалаврський) рівень вищої освіти

Спеціальність 113 Прикладна математика

Освітня програма  Комп’ютерне моделювання

  та технології програмування

Виконавець:

Студент групи ПА–21–1

Меляхов Максим

Варіант № 10

Дніпро

2023

* Постановка завдання

Застосувати патерн Компонувальник (Composite) для задачі обчислення похідної довільної складеної функції. Використати дані табл.1.2

Варіант індивідуального завдання 10:



Інтерфейс «Function»

Це інтерфейс у Java проєкті. Він визначає три методи:

* «calculate»: цей метод приймає значення double x і повертає значення типу double, яке представляє результат обчислення функції на вхідних даних x.
* «derivative»: цей метод повертає новий об'єкт Function, який представляє похідну вихідної функції.
* «toPrettyString»: цей метод приймає об’єкт «NumberFormat» і повертає рядкове представлення функції.

Інтерфейс у Java — це набір сигнатурних методів (без реалізацій), які можуть реалізувати інші класи. Це дозволяє використовувати поліморфізм, коли об’єкти різних класів, які реалізують той самий інтерфейс, можна розглядати так, ніби вони одного типу.

«NumberFormat» — це клас у Java, який використовується для форматування чисел відповідно до певної мови. Він є частиною пакета java.text і надає набір методів для форматування та аналізу числових значень у локалізовані рядки.

Клас Java «Const».

Цей клас реалізує два інтерфейси, «Number» і «Function».

Клас має три статичні об’єкти: «ZERO», «ONE» і «NEGATIVE\_ONE». Вони представляють константні значення 0, 1 і -1 відповідно.

Клас створює приватну змінну «value» типу double.

Клас оголошує конструктор який приймає значення типу double та присвоює його змінній «value».

Клас реалізує методи «calculate», «derivative» і «toPrettyString» з інтерфейсу «Function».

* «calculate»: цей метод повертає константне значення змінної «value»;
* «derivative»: цей метод повертає об’єкт «ZERO», оскільки похідна константи завжди дорівнює 0.
* «toPrettyString»: цей метод приймає об’єкт «NumberFormat» і повертає відформатоване представлення константного значення у вигляді рядка.

Також у класі реалізовано методи «intValue», «longValue», «floatValue» та «doubleValue» з інтерфейсу «Number». Ці методи повертають значення константи типу int, long, float, double відповідно.

Загалом, клас «Const» забезпечує просту реалізацію константної функції в Java, що дозволяє легко створювати та маніпулювати константними значеннями в математичних обчисленнях.

Клас Java «Linear»

Цей клас реалізує інтерфейс «Function».

Клас має один статичний об’єкт «X». Він представляє лінійну функцію y = x.

Клас створює приватну змінну «coefficient» типу double, яка зберігає коефіцієнт лінійної функції.

Клас має конструктор, який отримує значення типу double для коефіцієнта і присвоює його змінній «coefficient».

Клас реалізує методи «calculate», «derivative» і «toPrettyString» з інтерфейсу «Function».

* «calculate»: цей метод отримує значення x типу double і повертає значення лінійної функції, яке дорівнює x помноженому на коефіцієнт.
* «derivative»: цей метод повертає новий екземпляр класу «Const», що представляє константний коефіцієнт лінійної функції, оскільки похідна лінійної функції є константою.
* «toPrettyString»: цей метод отримує об'єкт типу NumberFormat і повертає відформатоване представлення лінійної функції у вигляді рядка. Відформатований рядок має вигляд "a\*x", де «a» - коефіцієнт лінійної функції.

Загалом, клас «Linear» забезпечує просту реалізацію лінійної функції на Java, що дозволяє легко створювати та маніпулювати лінійними функціями в математичних розрахунках.

Абстрактний клас «Composite»

Він реалізує інтерфейс "Function" і має ArrayList об'єктів "Function" з назвою "terms".

Клас позначено як абстрактний, що означає, що його не можна безпосередньо створити.

Клас має загальнодоступний метод доступу з іменем "terms", який повертає масив ArrayList об'єктів типу Function.

Клас має три конструктори:

* Перший конструктор ініціалізує масив ArrayList об'єктів типу Function пустим ArrayList.
* Другий та третій конструктори отримують на вхід масив об'єктів типу Function та ініціалізують ArrayList об'єктів типу Function новим ArrayList, що містить елементи вхідного масиву.

У класі не реалізовано методи "calculate", "derivative" та "toPrettyString" інтерфейсу "Function". Це означає, що будь-який клас, який розширює клас Composite, може реалізувати ці методи.

Загалом, клас "Composite" забезпечує базову структуру для створення складених функцій у Java, дозволяючи легко керувати списком об'єктів класу "Function". Класи, які розширюють цей клас, можуть додавати функціональність для представлення більш складних складених функцій.

Клас "Sum"

Цей клас розширює абстрактний клас "Composite" і реалізує інтерфейс "Function".

Клас має три конструктори:

* Перший конструктор ініціалізує ArrayList об'єктів "Function" порожнім ArrayList.
* Другий та третій конструктори отримують масив об'єктів "Function" та ініціалізують ArrayList об'єктів "Function" новим ArrayList, що містить елементи вхідного масиву.

У класі реалізовано методи "calculate", "derivative" та "toPrettyString" інтерфейсу "Function".

* "calculate": цей метод обчислює суму значень, повернутих методом "calculate" кожного об'єкту "Function" в ArrayList.
* "derivative": цей метод повертає новий об'єкт "Sum", який містить похідну кожного об'єкта "Function" у списку ArrayList.
* "toPrettyString": цей метод повертає рядкове представлення об'єкта "Sum" за допомогою методу "toPrettyString" кожного об'єкта "Function" у списку ArrayList. Доданки відокремлюються знаками "+", а вся сума береться в круглі дужки. Якщо доданок має від'ємний коефіцієнт, то знак "+" замінюється на знак "-".

Клас "Sum" представляє собою суму об'єктів "Function", що дозволяє легко компонувати функції. Клас надає функціональність для обчислення суми, знаходження похідної суми та повернення рядкового представлення суми.

Клас "Multiplication"

Цей клас реалізує інтерфейс Function і представляє добуток двох функцій. Він має два приватні поля, first і second, в яких зберігаються функції, що перемножуються.

У класі реалізовано методи "calculate", "derivative" та "toPrettyString" інтерфейсу "Function".

* "calculate": цей метод повертає добуток результатів обчислення першої та другої функцій для заданого вхідного значення x.
* "derivative": цей метод повертає похідну від добутку двох функцій, використовуючи правило диференціювання за добутком. Це передбачає створення двох нових об'єктів Multiplication з похідними першої та другої функцій відповідно, а потім їх додавання за допомогою класу Sum.
* "toPrettyString": цей метод повертає рядкове представлення операції множення двох функцій

Клас "Multiplication" представляє собою добуток об'єктів "Function", що дозволяє легко компонувати функції. Клас надає функціональність для обчислення добутка, знаходження похідної добутка та повернення рядкового представлення добутка.

Клас " Power"

Цей клас є реалізацією інтерфейсу Function, який представляє степеневу функцію виду , де - інша функція, а - сталий показник степеня.

Клас має дві приватні змінні: fx, яка є функцією, що підноситься до степеня, та index, яка є значенням показника степеня.

Конструктор отримує об'єкт-функцію fx та значення index і відповідним чином ініціалізує змінні класу.

У класі реалізовано методи "calculate", "derivative" та "toPrettyString" інтерфейсу "Function".

* "calculate": цей метод обчислює результат степеневої функції, використовуючи метод Math.pow() для піднесення результату функції fx, для заданого вхідного значення x, до степеня index.
* "derivative": цей метод обчислює похідну степеневої функції за допомогою ланцюгового правила диференціювання. Він повертає об'єкт Multiplication об'єкта Const зі значенням показника степеня index, об'єкт Power з тією ж функцією fx і показником степеня index-1, а також похідну функції fx.
* "toPrettyString": цей метод повертає рядкове представлення степеневої функції у красивому форматі за допомогою наданого об'єкта NumberFormat.

Клас "Power" представляє собою степеневу функцію об'єкту "Function" та значення типу double. Клас надає функціональність для обчислення степеневої функції, знаходження її похідної та повернення її рядкового представлення.

Клас "Abs"

Цей клас реалізує інтерфейс Function. Клас отримує єдиний об'єкт Function як параметр у своєму конструкторі.

У класі реалізовано методи "calculate", "derivative" та "toPrettyString" інтерфейсу "Function".

* "calculate": цей метод повертає модуль вхідної функції для будь-якого заданого значення x.
* "derivative": цей метод повертає похідну функції модуля за допомогою ланцюгового правила диференціювання, що дає результат у вигляді композиції вхідної функції, її похідної та оберненої функції модуля.
* "toPrettyString": цей метод повертає рядкове представлення функції за допомогою методу toPrettyString() вхідної функції, оточивши його функцією "abs".

Клас "Abs" представляє собою функцію модуля об'єкту "Function". Клас надає функціональність для обчислення функції модуля, знаходження її похідної та повернення її рядкового представлення.

Клас " Cbrt"

Цей клас реалізує інтерфейс Function та отримує об'єкт Function у своєму конструкторі.

У класі реалізовано методи "calculate", "derivative" та "toPrettyString" інтерфейсу "Function".

* "calculate": цей метод повертає кубічний корінь із заданого значення функції для вказаного значення x.
* "derivative": цей метод повертає похідну кубічного кореня функції, яка обчислюється за допомогою ланцюгового правила диференціювання. Результатом є новий об'єкт Function, який представляє похідну вихідної функції.
* "toPrettyString": цей метод повертає рядкове представлення функції кубічного кореня за допомогою методу toPrettyString об'єкта Function, переданого йому як параметр.

Клас "Cbrt" представляє собою функцію кубічного кореня об'єкту "Function". Клас надає функціональність для обчислення функції кубічного кореня, знаходження її похідної та повернення її рядкового представлення.

Клас " Cos"

Цей клас реалізує інтерфейс Function та представляє функцію косинуса. Клас Cos отримує іншу функцію fx у своєму конструкторі.

У класі реалізовано методи "calculate", "derivative" та "toPrettyString" інтерфейсу "Function".

* "calculate": цей метод повертає функцію косинуса до результату fx.calculate(x), щоб обчислити значення функції Cos у точці x.
* "derivative": цей метод повертає похідну функції Cos. Похідна функції косинуса - це функція від'ємного синуса, помножена на похідну внутрішньої функції fx.
* "toPrettyString": цей метод повертає рядкове представлення функції Cos.

Клас "Cos" представляє собою функцію косинуса об'єкту "Function". Клас надає функціональність для обчислення функції косинуса, знаходження її похідної та повернення її рядкового представлення.

Клас " Sin"

Цей клас реалізує інтерфейс Function. Він має приватну зміну з іменем fx типу Function, яка ініціалізується за допомогою конструктора класу.

У класі реалізовано методи "calculate", "derivative" та "toPrettyString" інтерфейсу "Function".

* "calculate": цей метод повертає обчислення синуса для функції fx і вхідного значення х за допомогою методу Math.sin.
* "derivative": цей метод повертає добуток косинуса вхідного значення на похідну внутрішньої функції fx.
* "toPrettyString": цей метод повертає рядкове представлення функції sin з її внутрішньою функцією fx.

Клас "Sin" представляє собою функцію синуса об'єкту "Function". Клас надає функціональність для обчислення функції синуса, знаходження її похідної та повернення її рядкового представлення.

Клас " Cosh"

Цей клас реалізує інтерфейс Function. Він отримує об'єкт Function як параметр у своєму конструкторі і зберігає його у закритій змінній fx.

У класі реалізовано методи "calculate", "derivative" та "toPrettyString" інтерфейсу "Function".

* "calculate": цей метод повертає функцію гіперболічного косинуса до результату fx.calculate(x), щоб обчислити значення функції Cosh у точці x.
* "derivative": цей метод повертає новий об'єкт Function, який представляє похідну функції Cosh, використовуючи ланцюгове правило диференціювання. Він повертає добуток fx.derivative() і sinh(fx).
* "toPrettyString": цей метод повертає рядкове представлення функції гіперболічного косинуса.

Клас "Cosh" представляє собою функцію гіперболічного косинуса об'єкту "Function". Клас надає функціональність для обчислення функції гіперболічного косинуса, знаходження її похідної та повернення її рядкового представлення.

Клас " Sinh"

Цей клас реалізує інтерфейс Function. У конструкторі він отримує єдиний аргумент fx типу Function, який представляє функцію, гіперболічного синуса яку потрібно обчислити.

У класі реалізовано методи "calculate", "derivative" та "toPrettyString" інтерфейсу "Function".

* "calculate": цей метод повертає функцію гіперболічного синуса до результату fx.calculate(x), щоб обчислити значення функції Sinh у точці x.
* "derivative": цей метод повертає похідну функції, яка обчислюється як добуток похідної функції fx на гіперболічний косинус fx.
* "toPrettyString": цей метод повертає відформатоване рядкове представлення функції.

Клас "Sinh" представляє собою функцію гіперболічного синуса об'єкту "Function". Клас надає функціональність для обчислення функції гіперболічного синуса, знаходження її похідної та повернення її рядкового представлення.

Клас " Logn"

Цей клас реалізує інтерфейс Function. Клас Logn отримує об'єкт Function як аргумент конструктора.

У класі реалізовано методи "calculate", "derivative" та "toPrettyString" інтерфейсу "Function".

* "calculate": цей метод повертає функцію натурального логарифма до результату fx.calculate(x), щоб обчислити значення функції Logn у точці x.
* "derivative": цей метод повертає новий об'єкт Function, який представляє похідну функції Logn, отриману множенням похідної внутрішньої функції на зворотну їй функцію.
* "toPrettyString": цей метод повертає рядкове представлення функції, яке включає слово "log", за яким слідує рядкове представлення внутрішньої функції.

Клас "Logn" представляє собою функцію натурального логарифма об'єкту "Function". Клас надає функціональність для обчислення функції натурального логарифма, знаходження її похідної та повернення її рядкового представлення.

Клас " Sech"

Цей клас реалізує інтерфейс Function. Клас Sech має конструктор, який отримує об'єкт Function як аргумент і зберігає цю функцію у змінній fx.

У класі реалізовано методи "calculate", "derivative" та "toPrettyString" інтерфейсу "Function".

* "calculate": цей метод повертає значення гіперболічного секансу вхідної функції при заданому вхідному значенні x.
* "derivative": цей метод повертає новий об'єкт Function, який представляє похідну гіперболічної січної функції. Похідна дорівнює - , де - функція гіперболічного тангенса, sech(x) - функція гіперболічного секанса, а - похідна вхідної функції.
* "toPrettyString": цей метод повертає рядкове представлення функції гіперболічного секанса, використовуючи метод "toPrettyString" вхідної функції.

Клас "Sech" представляє собою функцію гіперболічного секанса об'єкту "Function". Клас надає функціональність для обчислення функції гіперболічного секанса, знаходження її похідної та повернення її рядкового представлення.

Клас " Tanh"

Цей клас реалізує інтерфейс Function. Конструктор класу "Tanh" отримує в якості аргументу об'єкт "Function".

У класі реалізовано методи "calculate", "derivative" та "toPrettyString" інтерфейсу "Function".

* "calculate": цей метод повертає значення гіперболічної дотичної в заданій точці за допомогою функції "tanh" з класу "Math".
* "derivative": цей метод повертає новий об'єкт "Function", який представляє похідну гіперболічної дотичної. Похідна дорівнює , де - гіперболічна функція дотичної, а - похідна вхідної функції.
* "toPrettyString": повертає рядкове представлення функції гіперболічного тангенса, використовуючи метод "toPrettyString" вхідної функції.

Клас "Tanh" представляє собою функцію гіперболічного тангенса об'єкту "Function". Клас надає функціональність для обчислення функції гіперболічного тангенса, знаходження її похідної та повернення її рядкового представлення.

"Main"

Це програма на Java, яка обчислює і відображає значення двох математичних функцій, їхні похідні та значення в заданій точці x.

Перша функція визначається за допомогою комбінації різних математичних функцій, таких як логарифм, абсолютне значення, гіперболічний тангенс, кубічний корінь, косинус і лінійна функція. Друга функція визначається за допомогою комбінації лінійної, синусоїдальної гіперболічної, абсолютної величини, косинуса і кубічного кореня.

Програма пропонує користувачеві ввести значення x за допомогою об'єкта сканера. Введене значення потім використовується для обчислення значень двох функцій, їхніх похідних і виведення їх на консоль за допомогою класу NumberFormat для форматування виводу.

Результати роботи програми та деталі її застосування.

Тестові приклади:



f\_1(x) = (cbrt(log(abs(tanh(-11\*x))))+(-1 \* (cos(abs((0,5\*x-11))))^3))

f\_1'(x) = ((((3 \* cbrt((log(abs(tanh(-11\*x))))^2)))^-1 \* ((abs(tanh(-11\*x)))^-1 \* ((tanh(-11\*x) \* (abs(tanh(-11\*x)))^-1) \* ((sech(-11\*x))^2 \* -11))))+((0 \* (cos(abs((0,5\*x-11))))^3)+(-1 \* ((3 \* (cos(abs((0,5\*x-11))))^2) \* ((-1 \* sin(abs((0,5\*x-11)))) \* (((0,5\*x-11) \* (abs((0,5\*x-11)))^-1) \* (0,5+0)))))))

f\_1(1.0) = 0,106712

f\_1'(1.0) =0,304432

f\_2(x) = (x \* (sinh(abs((-11 \* cbrt(cos(0,5\*x))))))^2)

f\_2'(x) = ((1 \* (sinh(abs((-11 \* cbrt(cos(0,5\*x))))))^2)+(x \* ((2 \* (sinh(abs((-11 \* cbrt(cos(0,5\*x))))))^1) \* (cosh(abs((-11 \* cbrt(cos(0,5\*x))))) \* (((-11 \* cbrt(cos(0,5\*x))) \* (abs((-11 \* cbrt(cos(0,5\*x)))))^-1) \* ((0 \* cbrt(cos(0,5\*x)))+(-11 \* (((3 \* cbrt((cos(0,5\*x))^2)))^-1 \* ((-1 \* sin(0,5\*x)) \* 0,5)))))))))

f\_2(1.0) = 351117108,808080

f\_2'(1.0) =-322251083,143319

Перевірка функцій:









* Висновки за результатами роботи.

Було набуто навичок використання абстракції даних на прикладі шаблону проектування патерну Компонувальник (Composite).