КОЛЛЕДЖ АВТОНОМНОЙ НЕКОММЕРЧЕСКОЙ ОБРАЗОВАТЕЛЬНОЙ ОРГАНИЗАЦИИ

ВЫСШЕГО ОБРАЗОВАНИЯ «НАУЧНО-ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ «СИРИУС»

(АНО ВО КОЛЛЕДЖ АНО ВО «УНИВЕРСИТЕТ «СИРИУС»)

**РЕФЕРАТ**

ПО ДИСЦИПЛИНЕ «ВЕДЕНИЕ В СПЕЦИАЛЬНОСТЬ»

на тему:

«МЕТОДЫ ДИНАМИЧЕСКОГО ПРОГРАММИРОВАНИЯ»

Выполнил студент группы «1.9.7.2»

Старцев Захар Вадимович

Проверил преподаватель: A.A.Тенигин

Федеральная территория «Сириус», 2022

Оглавление

[Введение 3](#_Toc116846697)

[Глава 1. История Динамического Программирования 4](#_Toc116846698)

[Глава 2. Основные понятия и обозначения Динамического Программирования 5](#_Toc116846699)

[Глава 3. Идея Динамического Программирования 8](#_Toc116846700)

[Глава 4. Решение типовых задач 9](#_Toc116846701)

[Глава 4. Динамика и матрица переходов 12](#_Toc116846702)

[Заключение 13](#_Toc116846703)

[Использованная Литература 14](#_Toc116846704)

# Введение

На данный момент много компаний в своей работе встречаются с математическими моделями. Математическая модель — своего рода система уравнения, неравенств, а так же формул и выражений. Которые описывают поведение реального объекта составляющих его характеристики взаимосвязи между ними. Действие построение математической модели называют математическим моделированием. Благодаря математическим моделям и моделированию экономического объекта, помогает увидеть анализ производственных процессов и эффективных принятие решений.

В своем докладе я бы хотел рассказать про Методы решений данных экономических задач через динамическое программирование и в целом про зарождения про такое понятие как динамическое программирование

# Глава 1. История Динамического Программирования

Термин «динамическое программирование» был введён в сороковых годах прошлого века Ричардом Беллманом, он был нужен для описания задач которых не решить без ответа на предыдущую задачу. Позднее он развил определение этого термина до современного уровня. Изначально эта сфера начиналась как системный анализ и инжиниринг, и её признал IEEE (институт инженеров электротехники и электроники). Имя Беллмана и его вклад в динамическое программирование увековечено в термине «уравнение Беллмана», которое стало базой теории динамического программирования. Это уравнение даёт формулировку оптимизационной задачи в рекурсивной форме. Следует, однако заметить, что термин «программирование» в сочетании слов «динамическое программирование» на самом деле к программированию, как написанию кодов программы, почти не относится, а обладает смысловым значением словосочетания «математическое программирование», являющееся аналогом оптимизации. Поэтому термин «программа» в этом смысле больше подразумевает оптимальную очерёдность операций для нахождения решения задачи.

# Глава 2. Основной Принцип Динамического Программирования

Мы разобрались с историей данного термина,теперь можем поговорить как он работает на 2 терминах если мы поймем концепцию то станет просто понять суть динамического программирования . И это перекрывающиеся подзадачи, либо оптимальный подструктуры

Перекрывающиеся подзадачи-- описывают сложные уравнения, которые при разбиении на более мелкие наборы уравнений повторно используют части более мелких уравнений более одного раза для получения ответа. Например, математическое уравнение, предназначенное для вычисления всех возможных результатов с использованием набора чисел, может вычислять один и тот же результат много раз, в то время как другие результаты вычисляются только один раз. Динамическое программирование сообщило бы этой задаче, что после вычисления результата в первый раз оно должно сохранить этот результат и позже подставить ответ в уравнение вместо того, чтобы вычислять его снова. При работе с длинными сложными процессами и уравнениями это экономит время и создает более быстрое решение, используя гораздо меньше шагов.

Оптимальные подструктуры--- создают решение, находя лучший ответ на все подзадачи, а затем создавая лучший общий ответ. Разбив сложную проблему на более мелкие, компьютер использует математическую систему, чтобы определить лучший ответ для каждой задачи. Он вычисляет ответ исходной задачи из меньших ответов. В этом процессе действительно есть недостатки. Хотя он дает решение, которое работает лучше всего с математической точки зрения, оно может быть или не быть лучшим решением в реальной жизни, в зависимости от типа проблемы и того, как она соотносится с реальным миром.

# Глава 3. Идея Динамического Программирования

В отличие от определенного синтаксиса кодирования или шаблонов проектирования, динамическое программирование — это не конкретный алгоритм, а способ мышления. Поэтому техника принимает множество форм, когда дело доходит до реализации.

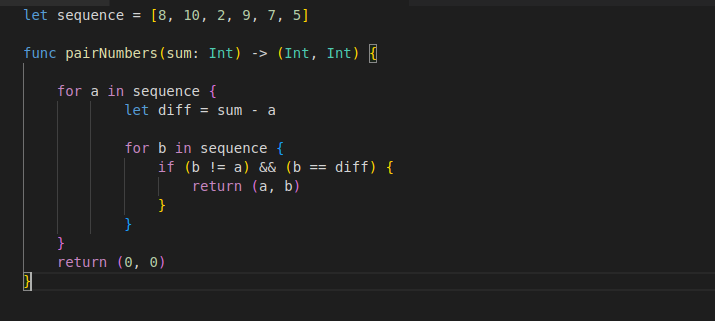
Основная идея динамического программирования состоит в том, чтобы рассмотреть серьезную проблему и разбить ее на более мелкие индивидуальные компоненты. Когда дело доходит до реализации, оптимальные методы основаны на хранении данных и повторном использовании для повышения эффективности алгоритма. Как мы увидим, многие вопросы разработки программного обеспечения решаются с помощью различных форм динамического программирования. Хитрость заключается в том, чтобы распознать, когда оптимальные решения могут быть разработаны с использованием простой переменной или требуют сложной структуры данных или алгоритма.

Например, переменные кода можно считать элементарной формой динамического программирования. Как мы знаем, цель переменной — зарезервировать определенное место в памяти для значения, которое будет вызвано позже.

# Глава 4. Решение типовых задач

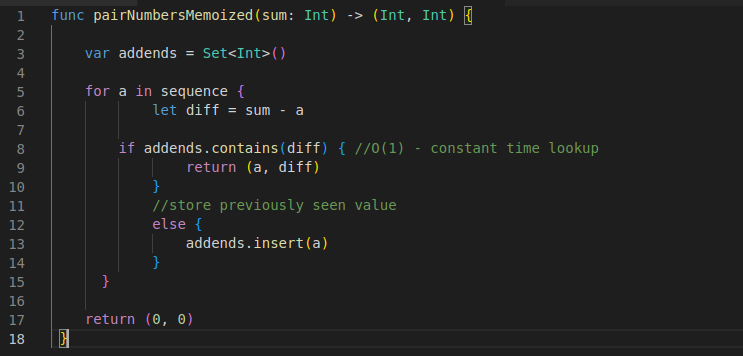
1 Грубая Сила

первый подход включает просмотр первого значения, а затем рассмотрение каждого последующего значения, чтобы определить, обеспечит ли оно разницу, необходимую для решения вопроса. Например, как только наш алгоритм проверит значение первого элемента массива, 8, он затем просканирует оставшиеся значения на наличие 3 (например, 11 — 8 = 3). Однако мы видим, что значение 3 не существует, поэтому алгоритм будет повторять тот же процесс для следующего значения (в нашем случае 10), пока не найдет успешную совпадающую пару. Не вдаваясь в подробности нотации большого O, мы можем предположить, что среднее время выполнения этого типа решения будет равно O(n ^ 2) времени или больше, главным образом потому, что наш алгоритм работает путем сравнения каждого значения с любым другим значением. В коде это можно представить следующим образом:



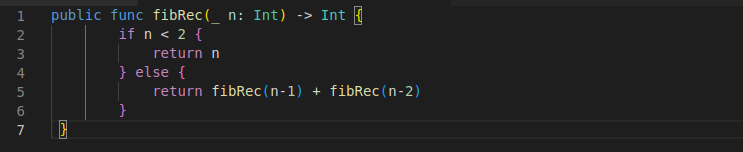
2)Запоминающейся подход ,

другой подход, использующий идею мемоизации. Прежде чем внедрять наш код, мы можем провести мозговой штурм, как сохранение ранее просмотренных значений поможет упростить процесс. Хотя использование стандартного массива возможно, объект набора наборов (также называемый хэш-таблицей или хэш-картой) может предоставить оптимизированное решение.

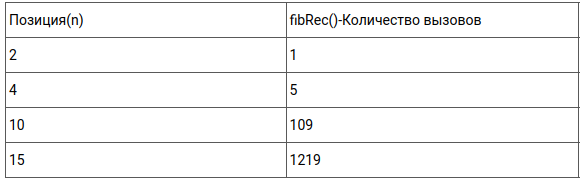


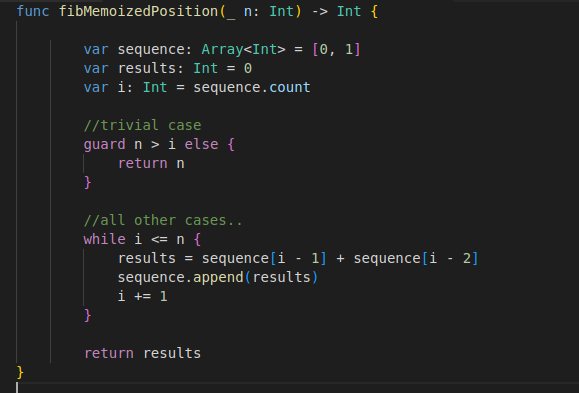
3) Последовательность Фибоначчи

При изучении различных методов программирования сразу приходит на помощь — рекурсия. Рекурсивные решения работают, имея модель, которая ссылается сама на себя. Таким образом, рекурсивные методы реализуются с помощью алгоритмов или структур данных. Хорошо известный пример рекурсии можно увидеть в последовательности Фибоначчи — числовой последовательности, полученной путем сложения двух предшествующих чисел (0, 1, 1, 2, 3, 5, 8, 13, 21 и т. д.):



при написание кода ошибок не возникло но мы так же должны обратить внимание на производительность данного алгоритма

Как показано, количество вызовов нашей функции значительно увеличилось. Как и в нашем предыдущем примере, производительность алгоритма снижается экспоненциально(когда скорость равна значению самой величины) в зависимости от размера входных данных. Это происходит потому, что операция не сохраняет ранее вычисленные значения. Без доступа к хранимым переменным единственный способ получить требуемые предшествующие значения — это рекурсия. Если предположить, что этот код используется в производственной среде, функция может привести к ошибкам или ошибкам производительности. Теперь мы можем реконструировать(перестроить) код для поддержки мемоизированного подхода:



# Глава 4. Минусы и Плюсы Динамического Программирования

В последней главе я бы хотел выделить все плюсы и минусы данного метода решения сложных задач, но все же я бы не хотел его называть шаблоном, Динамическое программирование это стиль мышления и восприятие новой информации а не просто зазубренная формула по которой можно решить несколько задач.

**Скорость**- для многих в решение поставленной задачи не мало важный фактор является скорость, динамическое программирование является одним из эффективнейших способов решений сложнейших задач за короткие сроки- нужно всего лишь заполнить данные таблички

**Универсальность-** в каждой задачи есть свой набор правил при решение и не имеют исключений и требуют минимальных подсчетов, в этом вопросе динамическое программирование справляется тоже отлично так как может выводить все возможные вариации решения задачи

**Точность**- ну и конечно итог каждый решенной(поставленной) задачи является правильность ответа и как я упоминал ранее Динамическое программирование может высчитывать все возможные варианты событий без каких либо погрешностей или неточностей

# Заключение

динамическое программирование — это не конкретный шаблон проектирования, а образ мышления. Его цель — создать решение для сохранения ранее просмотренных значений для повышения эффективности использования времени. динамическое программирование обеспечивает основу практически для всех программ. Это включает в себя использование простых переменных и сложных структур данных.

# Использованная Литература

1) https://habr.com/ru/post/113108/

2) https://www.youtube.com/watch?v=kWPVvrHEhLs&t=512s

3)https://www.youtube.com/watch?v=GOF4VUi4nGU&t=12s

4)https://gb.ru/blog/dinamicheskoe-programmirovanie/