Муниципальное бюджетное общеобразовательное учреждение «Лицей»

**Исследовательская работа**

«Разнообразие задач, решаемых методом динамического программирования»

**Выполнила:**

Старшова Ксения Павловна

Ученица 10«А» класса

**Руководитель:**

Белова Татьяна Владимировна учитель информатики МБОУ «Лицей»

г. Арзамас

2021

**Оглавление**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Введение** | **………………………………………………………………………………..1** | | | | | |
| **Метод динамического программирования** | | | | | | **……………………………………….2** |
| **Решение задания №18** | | | | **……………………………………………………………….5** | | |
| **Решение задания №23** | | | | **……………………………………………………………….8** | | |
| **Заключение** | | **……………………………………………………………………………9** | | | | |
| **Список используемой литературы** | | | | | **………………………………………………10** | |
| **Приложения** | | | **…………………………………………………………………………15** | | | |

**Введение**

**Цель:** применение метода динамического программирования при решении задач.

**Задача:** изучить и составить алгоритм применения метода динамического программирования в заданиях №18 и №23 ЕГЭ по информатике.

**Актуальность:** возможность применения метода динамического программирования для решения широкого ряда задач.

**Метод динамического программирования**

Метод динамического программирования – поиск решения сложных задач с помощью метода оптимизации, то есть разбиение на более простые задачи. Идеей этого метода является рассмотрения исходной сложной задачи как представителя сходных с ней задач. Он предусматривает значительное сокращение времени вычислений. При любом состоянии системы, следующий шаг выбирается так, чтобы выигрыш на данном шаге с оптимальным выигрышем на последующих шагах, был максимальный. Главным недостатком является проблема размерности - сложность вычислений возрастает при увеличении размерности задачи.

При решении задач используется подход мемоизации, для оптимизации рекурсии. Рекурсивные методы решения задач, как правило, производят полный перебор всех вариантов. Динамическое программирование также производит полный перебор всех вариантов, но при этом сохраняется результат уже решенной подзадачи и при необходимости он просто достается из памяти.

Динамическое программирование обычно придерживается двух подходов к решению задач:

* Восходящее динамическое программирование: заранее просчитываются все подзадачи, которые понадобятся для решения исходной задачи, а затем используются для построения решения исходной задачи. Этот способ выигрывает по размеру и количеству вызовов функций, но возникает сложность в выяснении, решении каких подзадач потребуется в дальнейшем.
* Нисходящее динамическое программирование: разбиение задачи на подзадачи меньшего размера, решение которых комбинируются для решения исходной задачи. Используется запоминание для решений уже решенных задач. Как правило, этот способ оказывается проще, если задача зависит не от одного параметра.

Задачи динамического программирования:

* Задача о вычислении чисел Фибоначчи
* Задача о выборе траекторий
* Задача о наибольшей общей последовательности
* Задача последовательного принятия решений
* Задача о рюкзаке
* Задача об использовании рабочей силы

Метод динамического подходит для решения широкого ряда задач. Для ещё большего ускорения решения задач можно использовать программы, такие как электронная таблица Excel и языки программирования. Такие задачи встречаются в ЕГЭ, где необходимо быстро и точно решить поставленную задачу.

**Решение задания №18**

Рассмотрим применение метода динамического программирования на примере решения задания №18 из варианта ЕГЭ по информатике. Суть задания в следующем:

*Квадрат разлинован на N×N клеток (1 < N < 17). Исполнитель Робот может перемещаться по клеткам, выполняя за одно перемещение одну из двух команд: вправо или вниз. По команде вправо Робот перемещается в соседнюю правую клетку, по команде вниз – в соседнюю нижнюю. При попытке выхода за границу квадрата Робот разрушается. Перед каждым запуском Робота в каждой клетке квадрата лежит монета достоинством от 1 до 100. Посетив клетку, Робот забирает монету с собой; это также относится к начальной и конечной клетке маршрута Робота. Определите максимальную и минимальную денежную сумму, которую может собрать Робот, пройдя из левой верхней клетки в правую нижнюю. В ответе укажите два числа – сначала максимальную сумму, затем минимальную. Исходные данные записаны в файле 18-0.xls в виде электронной таблице размером N×N, каждая ячейка которой соответствует клетке квадрата.*

Итак, открыв файл с таблицей, мы имеем следующие данные: Приложение №1

Начальное положение Робота в ячейке А1, конечное положение в J10. Робот может совершать движение только вправо или вниз. Наша задача сводиться к тому, чтобы пройти из начальной ячейки в конечную, собрав максимальную сумму монет. Построим ниже исходной таблицы таблицу, в которой будем рассчитывать сумму монет, которую мы можем собрать, посетив каждую клеточку. Итак, получаем: Приложение №2

Другими словами, нужно представить, что Робот заходит в определенную комнату и берет себе в мешок то количество монет, что есть в комнате. То есть в первой в мешочке у Робота будет лежать столько, сколько в исходной начальной ячейке. Дальше просчитаем клеточки верхней строки от B13до J13. Рассуждаем – в эти ячейки (комнаты) Робот может попасть только из комнат с левой стороны (так как направление движения он может совершать только вправо или вниз). Получаем, что количество взятых из этих комнат монет будет равно сумме количества монет из предыдущей комнаты и количества монет в этой комнате: Приложение №3

И растягиваем эту формулу в правую сторону до конца границы квадрата. Аналогично рассуждаем с ячейками диапазона A14:A22. Сумма собранных монет будет складываться из количества монет, собранных в комнате выше, и количества монет, которые находятся в текущей комнате. И растягиваем формулу вниз. После этого мы имеем следующий вид расчетной таблицы: Приложение №4

Теперь рассматриваем ячейки диапазона В4:J22. Они будут вычисляться следующим образом: в эти комнаты мы можем попасть либо из комнаты сверху, либо из комнаты слева. Нам важно выбрать оптимальный путь – то есть тот, который принесет нам максимальную сумму, значит, нам нужна комната, в которой Робот собрал наибольшую сумму. В этом случае нам поможет встроенная функция =МАКС( ), которая позволит выбрать наибольшее значение из ячеек. Получаем следующую формулу – берет количество монет, которые находятся в комнате плюс наибольшую сумму из двух ячеек – верхней и левой: Приложение №5

И растягиваем это формулу сначала вниз, потом вправо. Результат мы увидим в ячейке J22. Он равен 1204. Приложение № 6

Аналогично вычисляется минимальное значение суммы собранных монет Роботом. Разница в том, что будем использовать встроенную функцию МИН() вместо МАКС(), так наш оптимальный путь будет проходить через комнаты, в которых Робот собирает меньшую сумму монет. Таблица будет выглядеть следующим образом: Приложение №7

Рассмотрим еще одно задание, в котором есть дополнительное условие:

*Квадрат разлинован на N×N клеток (1 < N < 17). Исполнитель Робот может перемещаться по клеткам, выполняя за одно перемещение одну из двух команд: вправо или вниз. По команде вправо Робот перемещается в соседнюю правую клетку, по команде вниз – в соседнюю нижнюю. При попытке выхода за границу квадрата Робот разрушается. Перед каждым запуском Робота в каждой клетке квадрата записано число от 10 до 99.* ***Посетив клетку с нечетным значением, Робот увеличивает счет на 1; иначе увеличивает счёт на 2.***

*Определите максимальное и минимальное значение счета, который может собрать Робот, пройдя из левой верхней клетки в правую нижнюю. В ответе укажите два числа – сначала максимальную сумму, затем минимальную. Исходные данные представляют*  *собой электронную таблицу в файле 18-J1 размером N×N, каждая ячейка которой соответствует клетке квадрата.*

Проанализируем условие задачи, получается, что проходя по комнатам, Робот собирает не монеты, а очки. То есть, зайдя в комнату, если там четное количество монет, Робот получает 2 очка, если нечетное – то одно. Поэтому, для решения задачи нам нужно построить дополнительную таблицу, по которой и будет осуществлять расчет. Эта таблица будет содержать тот счет, который может получить Робот в каждой комнате. Воспользуемся встроенными функциями ЕСЛИ() и нам нужна будет функция вычисления остатка от деления одного числа на другое ОСТАТ(). С помощью нее мы определим четность числа. Ее значение в случае четного числа должно равняться 0: Приложение №8

И растягиваем формулу на весь диапазон ячеек от А12 до J21. Сначала по строчке вправо, потом вниз. Получаем таблицу значений счета: Приложение №9

А дальше рассуждаем аналогично предыдущей задачи, только расчет ведем уже не по исходным данным, а по таблице промежуточных вычислений – таблице, отражающей счет, который может пополнить Робот. Получаем: Приложение №10

Таким образом, метод динамического программирования можно применить для решения широкого ряда задач представленных в задании №18 ЕГЭ по информатике.

**Решение задания №23**

Рассмотрим применение метода динамического программирования на примере решения задания №23 из варианта ЕГЭ по информатике. Суть задания в следующем:

*У исполнителя Калькулятор три команды, которым присвоены номера:*

*1. прибавь 1*

*2. прибавь 2*

*3. умножь на 3*

*Сколько есть программ, которые число 1 преобразуют в число 12?*

Задача сводиться к тому, чтобы из числа 1 получить число 12, используя данные три команды. В задачах такого типа легче считать, сколько получиться программ, начиная с конца.

Для облегчения создания программы на языке программирования, необходимо составить рекуррентную формулу:

F(12)=F(11)+F(10)+F(4)

F(11)=F(10)+F(9)

F(10)=F(9)+F(8)

F(9)=F(8)+F(7)+F(3)

F(8)=F(7)+F(6)

F(7)=F(6)+F(5)

F(6)=F(5)+F(4)+F(2)

F(5)=F(4)+F(3)

F(4)=F(3)+F(2)

F(3)=F(2)+F(1)

F(2)=F(1)

F(1)=1

Где F(n) – функция получения числа n, использую команды исполнителя.

Проанализировав полученную рекуррентную формулу, можно заметить, что если n кратно 3, то формула записывается в виде: F(n)=F(n-1)+F(n-2)+F(n//3). В остальных случаях формула выглядит в виде: F(n)=F(n-1)+F(n-2).

Заметим, что используя команды исполнителя, можно получить числа меньше 1, которые не удовлетворяют условиям задачи и являются посторонними корнями. Поэтому необходимо задавать условия в программе.

Получаем общую формулу:

F(n)=0, если n<1

F(n)=1, если n=1

F(n)=F(n-1)+F(n-2)+F(n//3), если n кратно 3, n>1

F(n)=F(n-1)+F(n-2), если n не кратно 3, n>1

На основе данной формулы пишется программа с помощью основных функций def и if и других дополнительных на любом языке программирования. Получаем:

Приложение №11

def – объявление функции

if – условие

return – возвращение значения

n – конечное число

Все задания №23 ЕГЭ решаются аналогично приведенному примеру.

У исполнителя Калькулятор три команды, которым присвоены номера:

1. прибавь 2

2. прибавь 4

3. прибавь 5

Программа для исполнителя – это последовательность команд. Сколько существует программ, которые число 31 преобразуют в число 51?

Приложение №12

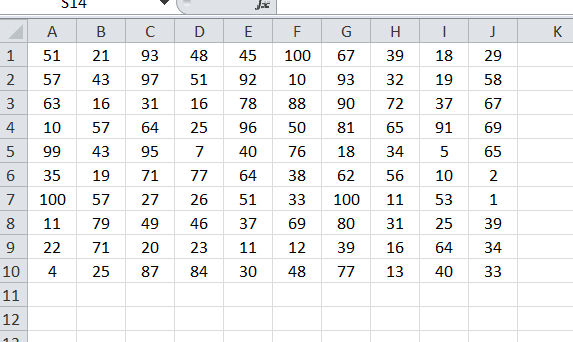
Таким образом, метод динамического программирования можно применить для решения задач представленных в задании №18 ЕГЭ по информатике.

**Заключение**

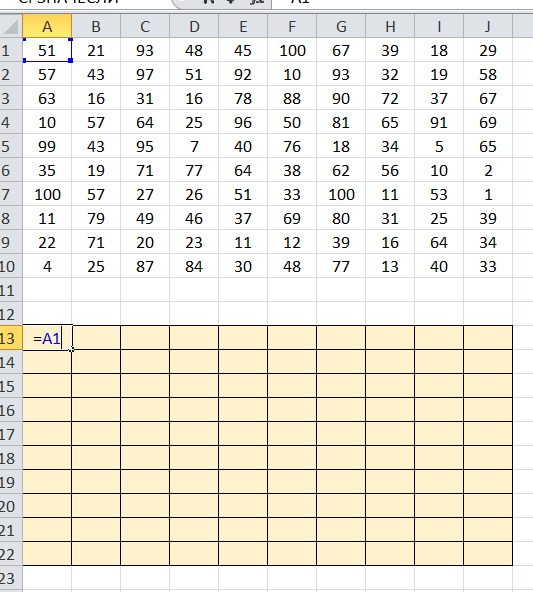
Динамическое программирование – метод поиска оптимального решения сложных задач путём разбиения их на более простые подзадачи, которые решаются наиболее рациональным и оптимальным способом. Этот метод позволяет значительно сократить время решения задач. Сфера использования данного метода чрезвычайно широка, круг задач, к которым он может быть применен, до настоящего времени ещё полностью не очерчен, поэтому включает большой спектр задач, в том числе задания №18 и №23 ЕГЭ.

**Приложения**

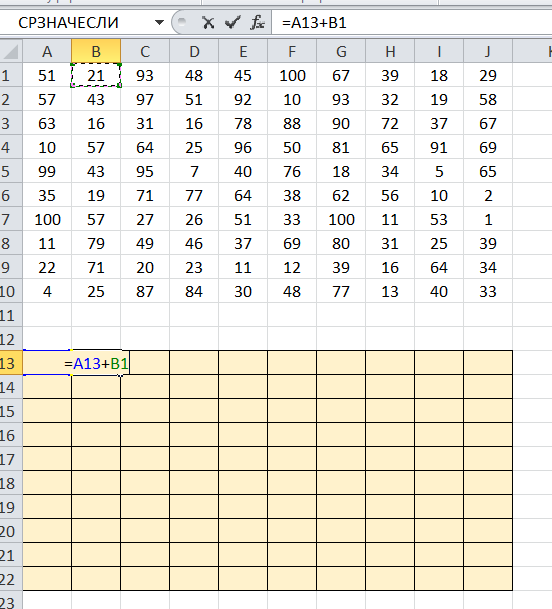
Приложение №1



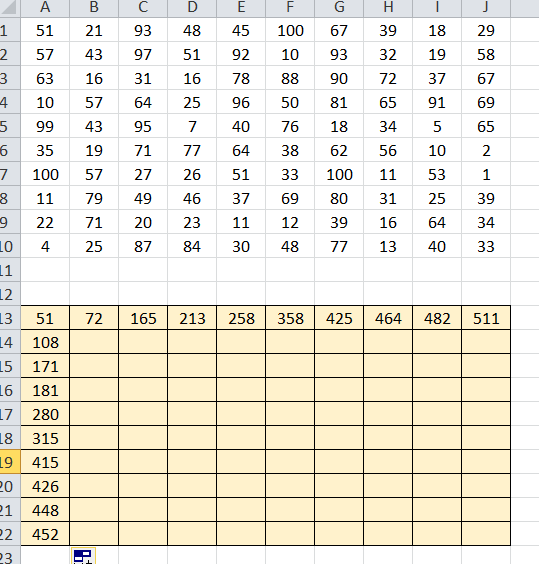
Приложение №2



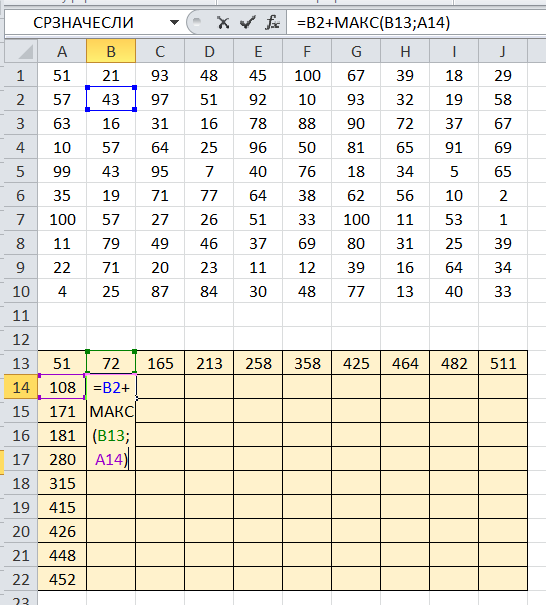
Приложение №3



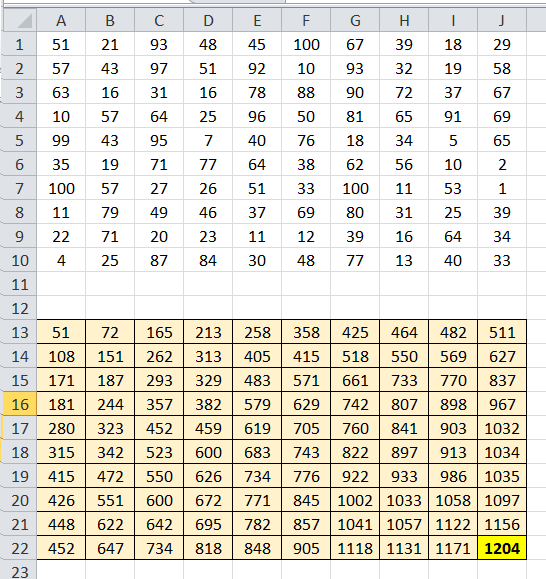
Приложение №4



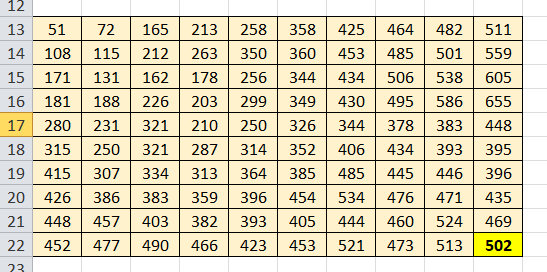
Приложение №5



Приложение №6



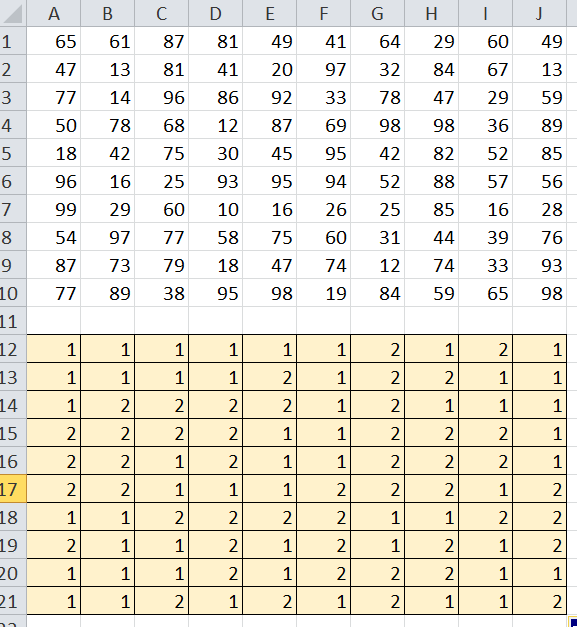
Приложение №7



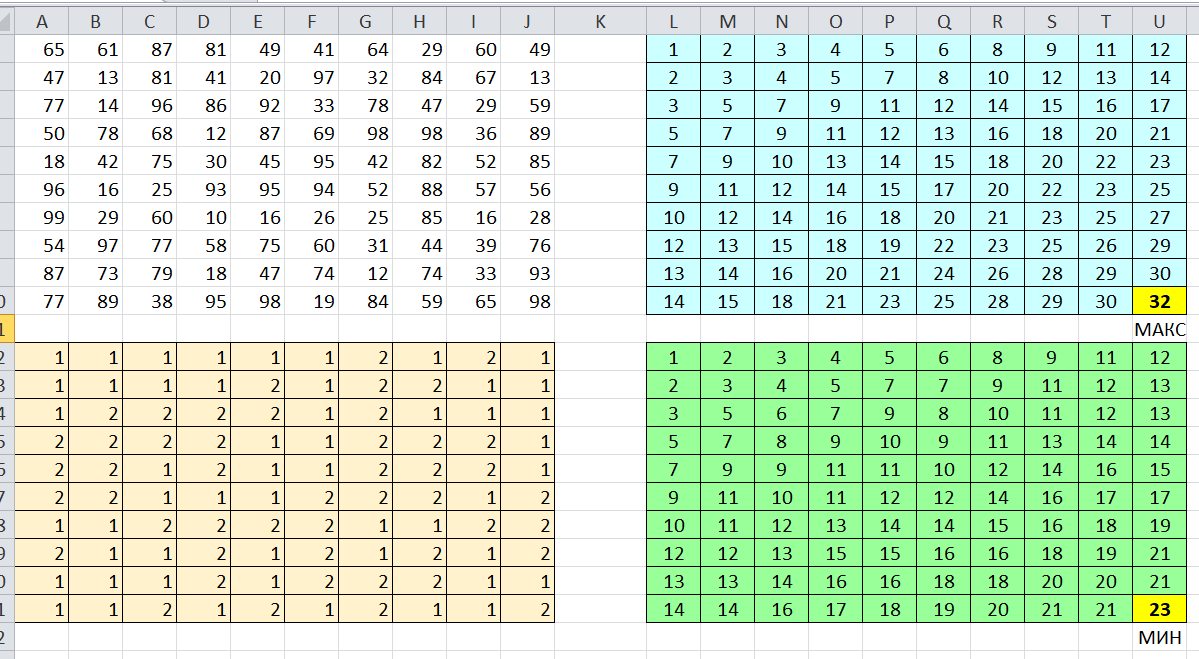
Приложение №8



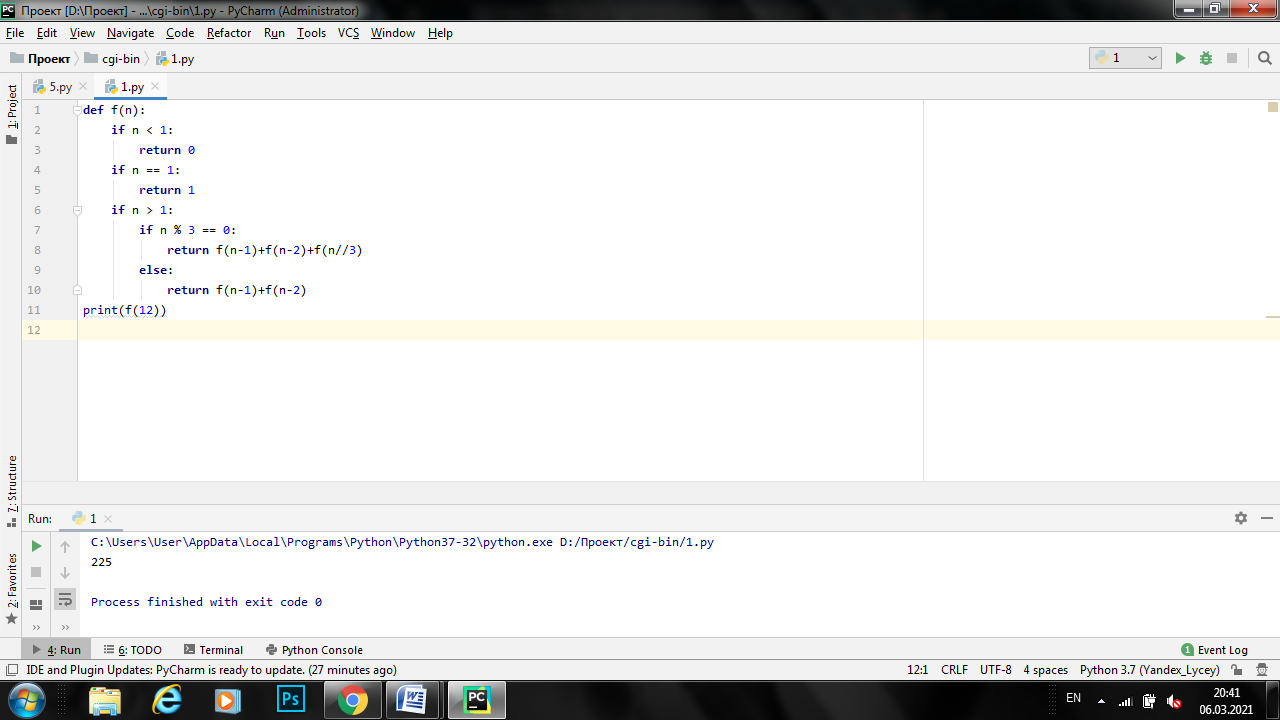
Приложение №9



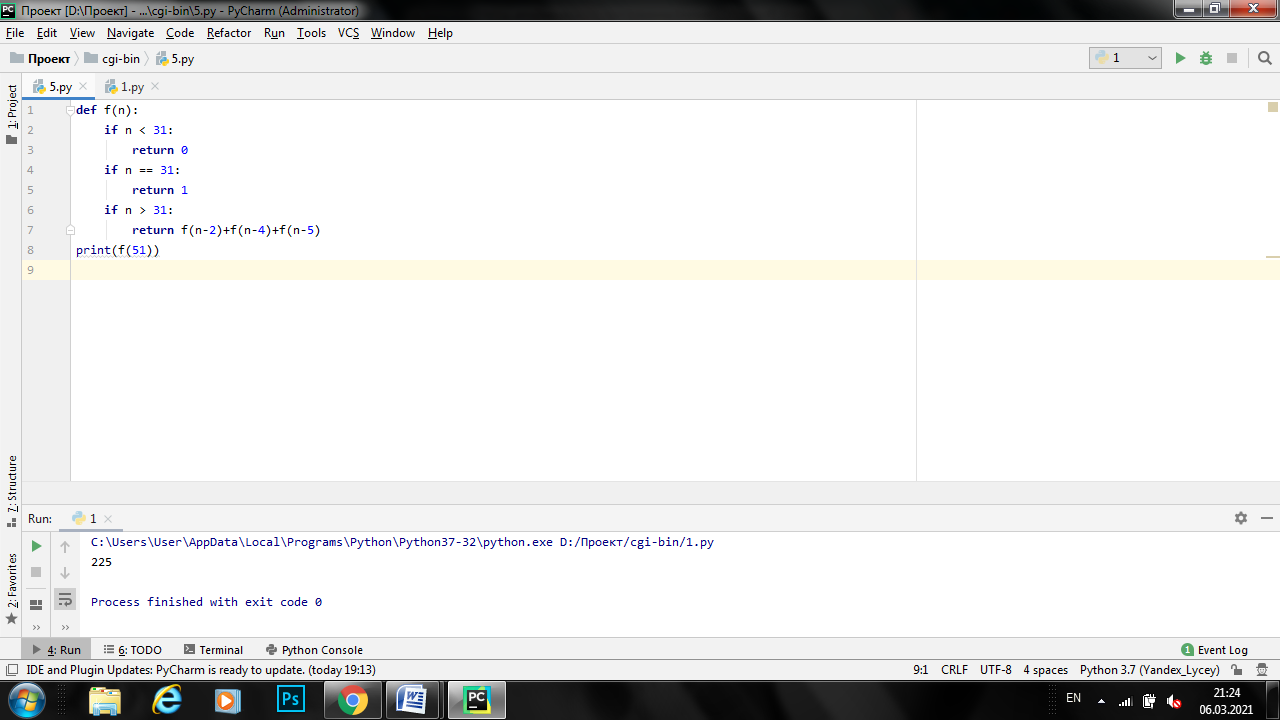
Приложение №10



Приложение №11



Приложение №12



**Литература**

1. [**http://www.math.mrsu.ru/text/courses/method/dinamicheskoe\_programmirovanie1.htm**](http://www.math.mrsu.ru/text/courses/method/dinamicheskoe_programmirovanie1.htm)
2. [**https://ru.wikipedia.org/wiki/**](https://ru.wikipedia.org/wiki/)
3. [**http://www.239.ru/sites/default/files/userdata/konspekt.\_dinamicheskoe\_programmirovanie.pdf**](http://www.239.ru/sites/default/files/userdata/konspekt._dinamicheskoe_programmirovanie.pdf)