Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Иркутский национальный исследовательский технический

университет»

Институт информационных технологий и анализа данных

**О Т Ч Ё Т**

|  |  |  |
| --- | --- | --- |
| о прохождении | | учебной практики |
|  | | (вид практики: учебная/производственная) |
| технологической (проектно-технологической) практики | | |
| (тип практики: технологическая/научно-исследовательская работа/преддипломная и др.) | | |
|  | | |
| в | ИРНИТУ | |
|  | (наименование профильной организации) | |

Обучающегося Василец В. Р., ИСИб-24-1

(ФИО, группа, подпись)

Руководитель практики от института ИТиАД

Кононенко Роман Владимирович, доцент

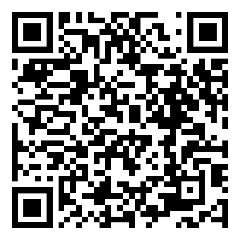
Ссылка на резюме https://www.superjob.ru/resume/programmist-s-55733621.html

(ФИО, должность, подпись)

Руководитель образовательной программы

Кононенко Р.В., доцент института ИТиАД

(ФИО, должность, подпись)

Оценка по практике

(ФИО, подпись, дата)

Содержание отчета на \_\_\_ стр. Приложение к отчету на \_\_\_ стр

Ссылка на резюме https://irkutsk.hh.ru/resume/b26a6c3eff0efde0e50039ed1f61686c6b4d49

Иркутск 2025

**Индивидуальное задание на прохождение**

**учебной практики: технологической (проектно-технологической) практики**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| для | Васильца Василия Романовича | | | | |
|  | (ФИО обучающегося полностью) | | | | |
| обучающегося | | 1 | курса | группы | ИСИб-24-1 |

по направлению подготовки Информационные системы и технологии

профиль Интеллектуальные системы обработки информации и управления

Место прохождения практики: ИРНИТУ

Сроки прохождения практики с «16» июня 2025 г. по «29» июня 2025 г.

Цели и задачи прохождения практики:

Содержание практики, вопросы, подлежащие изучению:

Планируемые результаты практики:

Руководитель практики от

института ИТиАД

\_\_\_\_\_\_\_\_\_\_\_/ Кононенко Р.В. /

(подпись

**Согласовано:**

Руководитель ООП

\_\_\_\_\_\_\_\_\_\_\_/Кононенко Р.В./

(подпись

«\_\_\_»\_\_\_\_\_\_\_\_\_\_2025 г.

С настоящим индивидуальным заданием и с программой практики ознакомлен, задание принято к исполнению

\_\_\_\_\_\_\_\_\_\_«16» июня 2025 г.

(подпись)

**ДНЕВНИК**

прохождения практики

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| обучающегося | | | | | Васильца Василия Романовича , ИСИб-24-1 |
|  | | | | | (фамилия, имя, отчество, группа) |
| курс | | 1 | | | |
| направление | | | | Информатика и вычислительная техника | |
| профиль | | | Интеллектуальные системы обработки | | |
| информации и управления | | | | | |
| в | ИРНИТУ | | | | |
|  | (наименование профильной организации)  Иркутск, 2025 | | | | |

Руководителем практики от структурного подразделения назначен:

Кононенко Роман Владимирович, доцент

(ФИО, должность)

**Рабочий график (план) прохождения практической подготовки**

(заполняется обучающимся)

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Период  практики | Содержание выполненных работ | Подпись руководителя практики от структурного подразделения |
| 1 | 16.06.2025 | Выполнены задачи №1, №2, №3 |  |
| 2 | 18.06.2025 | Создал резюме на сайтах hh.ru и superjob |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

|  |  |
| --- | --- |
| Дата фактического прибытия |  |
| обучающегося в структурное подразделение | 16.06.2026 |
| Дата фактического убытия |  |
| обучающегося из структурного подразделения | 28.06.2025 |

|  |  |
| --- | --- |
| Руководитель образовательной программы | Кононенко Р.В. |
|  | (ФИО, подпись) |
| Директор института | Говорков А.С. |
|  | (ФИО, подпись) |

**Задание №1**

Постановка задачи:

Незнайка в своей экспедиции на Луну оказался на вершине лунной горы. Спуск вниз опасен, поэтому он взял с собой карту склона горы, где числами обозначено, сколько минут требуется на этот участок маршрута. Спуск происходит сверху вниз на один из соседних участков. Напишите программу, рассчитывающую минимальное время спуска (сумму чисел в пути с вершины до основания).

Ход решения:

**import** **random**

**def** **time**(n, pyramid):

dp = [[**0**] \* (i + **1**) **for** i **in** range(n)]

dp[**0**][**0**] = pyramid[**0**][**0**]

**for** i **in** range(**1**, n):

**for** j **in** range(i + **1**):

**if** j == **0**:

dp[i][j] = dp[i - **1**][j] + pyramid[i][j]

**elif** j == i:

dp[i][j] = dp[i - **1**][j - **1**] + pyramid[i][j]

**else**:

dp[i][j] = min(dp[i - **1**][j - **1**], dp[i - **1**][j]) + pyramid[i][j]

min\_time = min(dp[n - **1**])

**return** min\_time, dp

**def** **path**(n, pyramid, dp):

min\_index = dp[n - **1**].index(min(dp[n - **1**]))

path = []

current\_index = min\_index

**for** i **in** range(n - **1**, -**1**, -**1**):

path.append(pyramid[i][current\_index])

**if** i > **0**:

**if** current\_index > **0** **and** dp[i - **1**][current\_index - **1**] < dp[i - **1**][current\_index]:

current\_index -= **1**

path.reverse()

**return** path

n = int(input())

pyramid = []

**for** i **in** range(n):

level = [random.randint(**1**, **100**) **for** \_ **in** range(i + **1**)]

pyramid.append(level)

**for** level **in** pyramid:

**print**(" ".join(map(str, level)))

time, dp = time(n, pyramid)

path = path(n, pyramid, dp)

**print**("Минимальное время пути:", time)

**print**("Сам путь", " ".join(map(str, path)))

Результат:



Рисунок 1 – пример выполнения программы 1

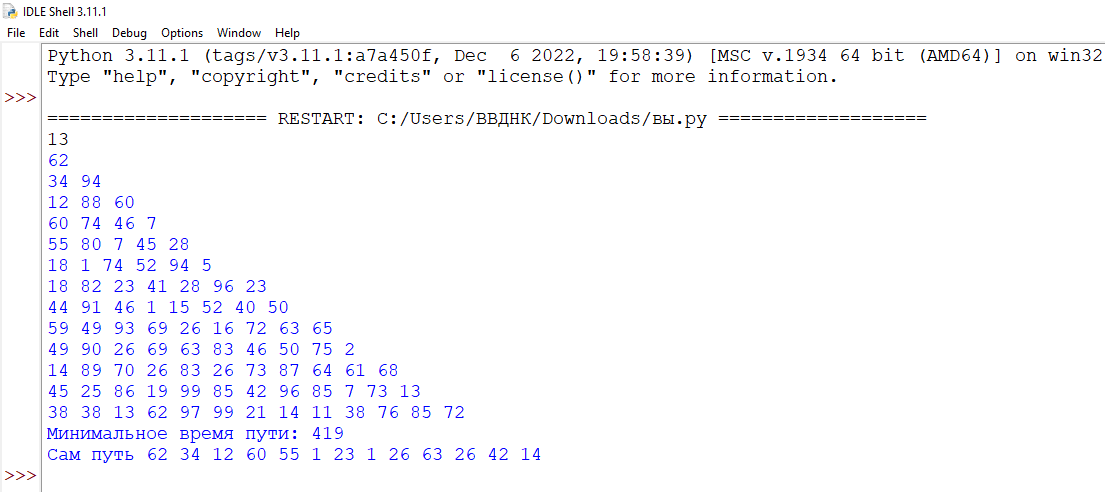


Рисунок 2 – пример выполнения программы 2

**Задание №2**

Постановка задачи:

После метеоритной атаки компьютерная сеть для управления лунными заводами разбилась на части, нужно объединить её в единое целое. Каждый фрагмент сети представлен в виде ненаправленного графа.

Вам известно общее число вершин графа (узлы сети, не более 1000) и набор рёбер (сохранившиеся линии связи, не более 1000).

Определите, какое минимальное число линий связи нужно дополнительно построить, чтобы сеть стала единой.

Ход решения:

**def** **components**(n, e):

graph = {i: [] **for** i **in** range(**1**, n + **1**)}

**for** u, v **in** e:

graph[u].append(v)

graph[v].append(u)

visited = set()

components = **0**

**def** **dfs**(node):

stack = [node]

**while** stack:

curr = stack.pop()

**for** neighbor **in** graph[curr]:

**if** neighbor **not** **in** visited:

visited.add(neighbor)

stack.append(neighbor)

**for** node **in** range(**1**, n + **1**):

**if** node **not** **in** visited:

components += **1**

visited.add(node)

dfs(node)

**return** components - **1**

n, m = map(int, input().split())

e = [tuple(map(int, input().split())) **for** \_ **in** range(m)]

**print**(components(n, e))

Результат:

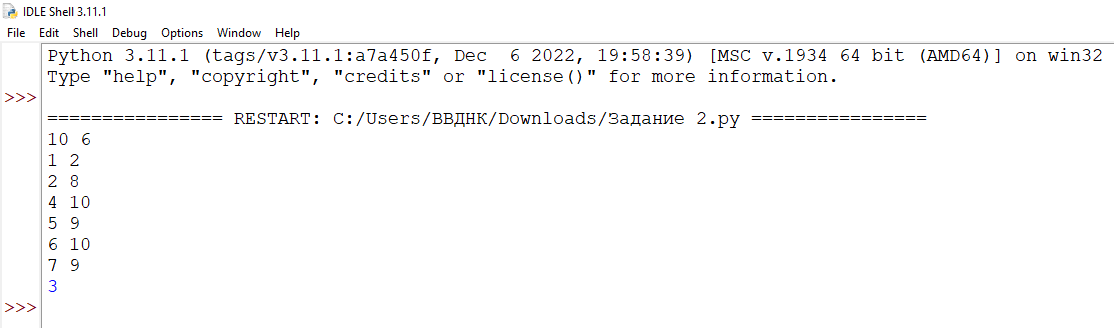


Рисунок 3 – пример выполнения программы 1

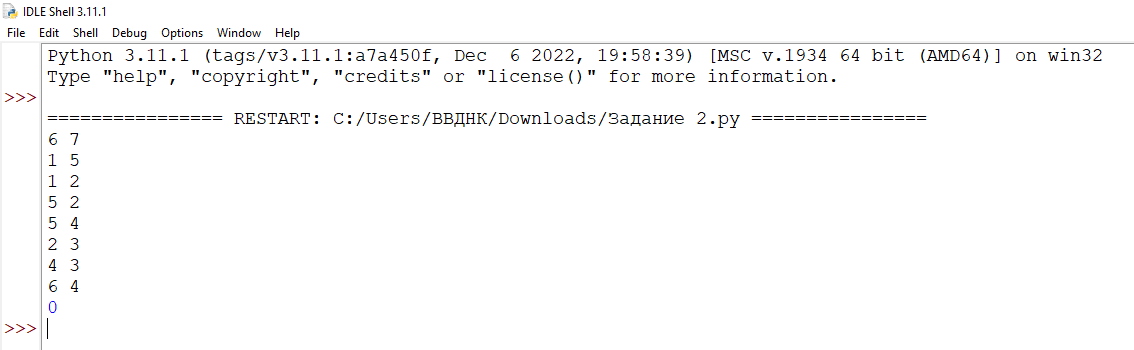


Рисунок 4 – пример выполнения программы 2

**Задача №3**

Постановка задачи:

В Иркутске раз в году наступает зима. Не смотря на то что событие это довольно регулярное, оно всегда внезапно. Снег буквально заваливает все улицы, не давая проехать на чём-то меньше трактора. В этом году терпение лопнуло и специальным указом был создан кризисный центр по борьбе с сугробами. Центру были переданы спутники, лазеры, метеорологические зонды и несколько десятков лопат.

Вам поручено возглавить отдел разведки снежной ситуации и быть способным чрезвычайно быстро отвечать на запросы центра. Сам город состоит из нескольких, расположенных подряд, улиц, каждая из которых абсолютна похожа на любую другую.

* Информация о снеге передается вам в виде тройки чисел – 1 в качестве идентификатора события, уникального индекса улицы и количество миллиметров выпавшего снега.
* Запросы в свою очередь так же имеют вид тройки чисел – 2 в качестве идентификатора события, индекс улицы с которой нужно суммировать количество выпавшего снега и индекс улицы по которую нужно суммировать, крайние улицы должны быть включены.

Ход решения:

**def** **snow**(n, k, events):

snow = [**0**] \* (n + **1**)

results = []

**for** event **in** events:

event\_type = event[**0**]

**if** event\_type == **1**:

street\_index = event[**1**]

snow\_amount = event[**2**]

snow[street\_index] += snow\_amount

**elif** event\_type == **2**:

start\_index = event[**1**]

end\_index = event[**2**]

total\_snow = sum(snow[start\_index:end\_index + **1**])

results.append(total\_snow)

**return** results

**if** \_\_name\_\_ == "\_\_main\_\_":

n, k = map(int, input().split())

events = [list(map(int, input().split())) **for** \_ **in** range(k)]

results = snow(n, k, events)

**for** result **in** results:

**print**(result)

Результат:

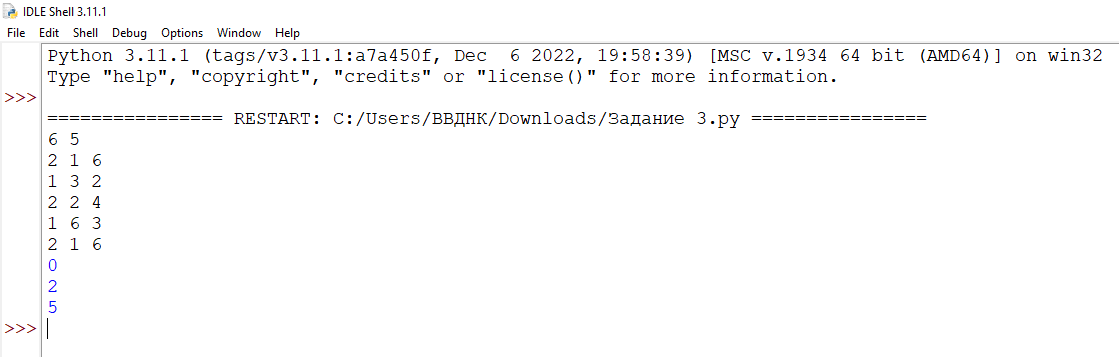


Рисунок 5 – пример выполнения программы 1

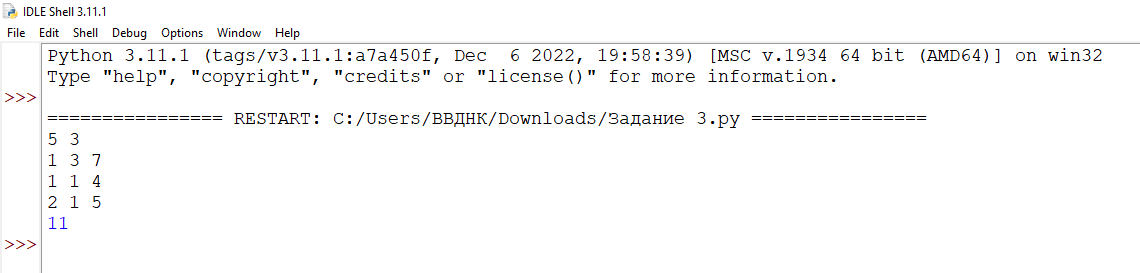


Рисунок 6 – пример выполнения программы 2