

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ВЛАДИВОСТОКСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «ВВГУ»)

ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И АНАЛИЗА ДАННЫХ
КАФЕДРА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И СИСТЕМ

ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №7

По дисциплине
«Информатика и программирование»

Студент		
гр. Бин-25-2	_____	И.А. Маклаков
Ассистент		
Преподавателя	_____	М.В. Водяницкий

Задание

Выполнить задания и написать отчёт по стандартам ВВГУ.

Задание 1. Имеется список объектов Фонда с указанием уровня угрозы:

```
objects = [  
    ("Containment Cell A", 4),  
    ("Archive Vault", 1),  
    ("Bio Lab Sector", 3),  
    ("Observation Wing", 2)  
]
```

Используя `sorted` и лямбда-выражение, отсортируйте объекты по возрастанию уровня угрозы

Задание 2. Дан список сотрудников Фонда с количеством проведенных смен и стоимостью одной смены:

```
staff_shifts = [  
    {"name": "Dr. Shaw", "shift_cost": 120, "shifts": 15},  
    {"name": "Agent Torres", "shift_cost": 90, "shifts": 22},  
    {"name": "Researcher Hall", "shift_cost": 150, "shifts": 10}  
]
```

Используя `map` и лямбда-выражение, создайте список общей стоимости работы каждого сотрудника

Затем найдите максимальную стоимость с помощью `max`

Задание 3. Дан список персонала с уровнем допуска:

```
personnel = [  
    {"name": "Dr. Klein", "clearance": 2},  
    {"name": "Agent Brooks", "clearance": 4},  
    {"name": "Technician Reed", "clearance": 1}  
]
```

Используя `map` и лямбда-выражение, создайте новый список, где каждому сотруднику добавляется категория допуска:

- "Restricted" - уровень 1
- "Confidential" - уровни 2–3
- "Top Secret" - уровень 4 и выше

Результат должен быть списком словарей

Задание 4. Дан список зон Фонда с указанием времени активности (в часах):

```
zones = [  
    {"zone": "Sector-12", "active_from": 8, "active_to": 18},  
    {"zone": "Deep Storage", "active_from": 0, "active_to": 24},  
]
```

```
{ "zone": "Research Wing", "active_from": 9, "active_to": 17 }  
]
```

Используя filter и лямбда-выражение, выберите зоны, которые полностью работают в дневной период (с 8 до 18 включительно)

Задание 5. Фонд анализирует служебные отчеты. Некоторые отчеты содержат внешние ссылки, которые должны быть удалены перед архивированием

```
reports = [  
    { "author": "Dr. Moss", "text": "Analysis completed. Reference: http://external-archive.net" },  
    { "author": "Agent Lee", "text": "Incident resolved without escalation." },  
    { "author": "Dr. Patel", "text": "Supplementary data available at https://secure-research.org" },  
    { "author": "Supervisor Kane", "text": "No anomalies detected during inspection." },  
    { "author": "Researcher Bloom", "text": "Extended observations uploaded to http://research-  
notes.lab" },  
    { "author": "Agent Novak", "text": "Perimeter secured. No external interference observed." },  
    { "author": "Dr. Hargreeve", "text": "Full containment log stored at https://internal-db.scp" },  
    { "author": "Technician Moore", "text": "Routine maintenance completed successfully." },  
    { "author": "Dr. Alvarez", "text": "Cross-reference materials: http://crosslink.foundation" },  
    { "author": "Security Officer Tan", "text": "Shift completed without incidents." },  
    { "author": "Analyst Wright", "text": "Statistical model published at https://analysis-hub.org" },  
    { "author": "Dr. Kowalski", "text": "Behavioral deviations documented internally." },  
    { "author": "Agent Fischer", "text": "Additional footage archived: http://video-storage.sec" },  
    { "author": "Senior Researcher Hall", "text": "All test results verified and approved." },  
    { "author": "Operations Lead Grant", "text": "Emergency protocol draft shared via https://ops-  
share.scp" }  
]
```

Используя filter и лямбда-выражение:

1. Отберите отчёты, содержащие ссылки (http или https)
2. Преобразуйте их так, чтобы вместо ссылки отображалось [ДАННЫЕ УДАЛЕНЫ]

Задание 6. Дан список SCP-объектов с указанием их класса содержания:

```
scp_objects = [  
    { "scp": "SCP-096", "class": "Euclid" },  
    { "scp": "SCP-173", "class": "Euclid" },  
    { "scp": "SCP-055", "class": "Keter" },  
    { "scp": "SCP-999", "class": "Safe" },  
    { "scp": "SCP-3001", "class": "Keter" }  
]
```

Используя filter и лямбда-выражение, сформируйте список SCP-объектов, которые требуют усиленных мер содержания

К объектам с усиленными мерами относятся все SCP, класс которых не равен "Safe"

Результат должен быть списком словарей исходного формата

Задание 7. Дан список инцидентов с количеством задействованного персонала:

```
incidents = [  
    {"id": 101, "staff": 4},  
    {"id": 102, "staff": 12},  
    {"id": 103, "staff": 7},  
    {"id": 104, "staff": 20}  
]
```

Используя sorted и лямбда-выражение:

1. Отсортируйте инциденты по количеству персонала
2. Оставьте только три наиболее ресурсоемких инцидента

Задание 8. Дан список протоколов безопасности и их уровней критичности:

```
protocols = [  
    ("Lockdown", 5),  
    ("Evacuation", 4),  
    ("Data Wipe", 3),  
    ("Routine Scan", 1)  
]
```

Используя map и лямбда-выражение, создайте новый список строк вида:

"Protocol Lockdown - Criticality 5"

Задание 9. Имеется список смен охраны с указанием длительности (в часах):

```
shifts = [6, 12, 8, 24, 10, 4]
```

Используя filter и лямбда-выражение, выберите только те смены, которые:

- длятся не менее 8 часов
- не превышают 12 часов

Задание 10. Дан список сотрудников с результатами психологической оценки (от 0 до 100):

```
evaluations = [  
    {"name": "Agent Cole", "score": 78},  
    {"name": "Dr. Weiss", "score": 92},  
    {"name": "Technician Moore", "score": 61},  
    {"name": "Researcher Lin", "score": 88}  
]
```

Используя \max и лямбда-выражение, определите сотрудника с наивысшей оценкой

Результатом должно быть имя сотрудника и его балл

Содержание

1	Выполнение работы	3
1.1	Задание 1	3
1.2	Задание 2	3
1.3	Задание 3	4
1.4	Задание 4	4
1.5	Задание 5	5
1.6	Задание 6	5
1.7	Задание 7	6
1.8	Задание 8	7
1.9	Задание 9	7
1.10	Задание 10	7

1 Выполнение работы

1.1 Задание 1

В данном задании сделан код, который сортирует объекты по возрастанию исходя из их уровня угрозы. Программа изображена на Рисунке 1.

```

1      objects = [
2          ("Containment Cell A", 4),
3          ("Archive Vault", 1),
4          ("Bio Lab Sector", 3),
5          ("Observation Wing", 2)
6      ]
7
8      sorted_ob = sorted(objects, key=lambda x: x[1])
9      print(*sorted_ob)

```

Рисунок 1 - Код для задания 1

Пояснение работы программы:

- 1) Из задания получили словарь объектов с их уровнями угрозы
- 2) Создаём переменную sorted_ob и даём ей значение через функцию sorted, она берёт данные из objects и сортирует их по второму элементу
- 3) Командой print() выводим полученную сортировку, символ *(звёздочки) «распаковывает» его в отдельные элементы и выводит через пробел

1.2 Задание 2

В данном задании сделан код, чтобы найти общую стоимость смен, что провёл каждый сотрудник. Программа изображена на Рисунке 2.

```

1      staff_shifts = [
2          {"name": "Dr. Shaw", "shift_cost": 120, "shifts": 15},
3          {"name": "Agent Torres", "shift_cost": 90, "shifts": 22},
4          {"name": "Researcher Hall", "shift_cost": 150, "shifts": 10}
5      ]
6
7      total = list(map(lambda x: x['shift_cost'] * x['shifts'], staff_shifts))
8      Max = max(total)
9
10     print(f'''
11     Total - {total}
12     Max - {Max}
13     ''')

```

Рисунок 2 - Код для задания 2

Пояснение работы программы:

- 1) Из задания получили список словарей: имя, стоимость смены, количество смен

2) Создали переменную `total` и для её значения применяем функцию `list()`, функцию `map()` и лямбда-выражение. В ней мы умножаем стоимость смены на их количество

3) В переменную `Max` записываем максимальное из списка `total` через функцию `max`

4) Выводим результат командой `print()`

1.3 Задание 3

В данном задании сделан код, который добавляет в словарь категорию допуска сотруднику из их уровня допуска. Программа изображена на Рисунке 3.

```

1 personnel = [
2     {"name": "Dr. Klein", "clearance": 2},
3     {"name": "Agent Brooks", "clearance": 4},
4     {"name": "Technician Reed", "clearance": 1}
5 ]
6
7 res = list(map(lambda x: {'name': x['name'], 'clearance': x['clearance'], 'category': 'Restricted' if x['clearance'] == 1
8                          else 'Confidential' if 2 <= x['clearance'] <= 3
9                          else 'Top Secret' if x['clearance'] >= 4
10                         else 'Broken Code'}, personnel))
11 print(res)

```

Рисунок 3 - Код для задания 3

Пояснение работы программы:

- 1) Дан список словарей сотрудника и его уровень допуска
- 2) Создаём переменную `res` и в ней будет новый список словарей с категорией допуска сотрудника через функции `list()`, `map()` и лямбда-выражение
- 3) В лямбда-выражении создаём словарь заново, используя то же имя, уровень допуска из списка словарей `personnel` и создаём условия, если уровень допуска равен 1 то категория «Restricted», если от 2 до 3, то категория «Confidential», если 4 и более, то категория «Top Secret»
- 4) Командой `print()` выводим результат

1.4 Задание 4

В данном задании сделан код, фильтрует и отображает зоны которые работают с 8 до 18 включительно. Программа изображена на Рисунке 4.

```

1 zones = [
2     {"zone": "Sector-12", "active_from": 8, "active_to": 18},
3     {"zone": "Deep Storage", "active_from": 0, "active_to": 24},
4     {"zone": "Research Wing", "active_from": 9, "active_to": 17}
5 ]
6
7 DayZone = list(filter(lambda zone: zone['active_from'] <= 8 and zone['active_to'] >= 18, zones))
8 print(DayZone)

```

Рисунок 4 - Код для задания 4

Пояснение работы программы:

- 1) Получаем список словарей из задания

2) Создаём переменную DayZone и придаём ей значения через функции list(), filter() и лямбда-выражение

3) В лямбда-выражении смотрим чтобы попали зоны только со значениями «активны от» 8 и меньше, и «активны до» до 18 и больше.

4) Выводим результат командой print()

1.5 Задание 5

В данном задании сделан код, который отбирает служебные отчёты, если они содержат ссылку http или https, то ссылку заменяет на [ДАННЫЕ УДАЛЕНЫ]. Программа изображена на Рисунке 5.

```

1  # -*- coding: cp1251 -*-
2  import re
3  reports = [
4      {"author": "Dr. Moss", "text": "Analysis completed. Reference: http://external-archive.net"},
5      {"author": "Agent Lee", "text": "Incident resolved without escalation."},
6      {"author": "Dr. Patel", "text": "Supplementary data available at https://secure-research.org"},
7      {"author": "Supervisor Hane", "text": "No anomalies detected during inspection."},
8      {"author": "Researcher Bloom", "text": "Extended observations uploaded to http://research-notes.lab"},
9      {"author": "Agent Novak", "text": "Perimeter secured. No external interference observed."},
10     {"author": "Dr. Hargreeve", "text": "Full containment log stored at https://internal-db.scp"},
11     {"author": "Technician Moore", "text": "Routine maintenance completed successfully."},
12     {"author": "Dr. Alvarez", "text": "Cross-reference materials: http://crosslink.foundation"},
13     {"author": "Security Officer Tan", "text": "Shift completed without incidents."},
14     {"author": "Analyst Wright", "text": "Statistical model published at https://analysis-hub.org"},
15     {"author": "Dr. Kowalski", "text": "Behavioral deviations documented internally."},
16     {"author": "Agent Fischer", "text": "Additional footage archived: http://video-storage.sec"},
17     {"author": "Senior Researcher Hall", "text": "All test results verified and approved."},
18     {"author": "Operations Lead Grant", "text": "Emergency protocol draft shared via https://ops-share.scp"},
19 ]
20
21 filtered = list(filter(lambda r: 'http' in r['text'] or 'https' in r['text'], reports))
22 res = list(map(lambda r: {'author': r['author'], 'text': re.sub(r'https?://[^\s]+', '[ДАННЫЕ УДАЛЕНЫ]', r['text'])}, filtered))
23 print(res)

```

Рисунок 5 - Код для задания 5

Пояснение работы программы:

- 1) Дан список словарей отчётов сотрудников
- 2) Импортируем библиотеку re для работы с регулярными выражениями
- 3) Создаём переменную filtered, для её значения использует функции list(), filter() и лямбда-выражение
- 4) В неё мы собираем все отчёты что содержат ссылку
- 5) Создаём переменную res и там через функции list(), map(), лямбда-выражение и библиотеку re заменяем ссылки на [ДАННЫЕ УДАЛЕНЫ]
- 6) Выводим результат командой print()

1.6 Задание 6

В данном задании сделан код, который отбирает объекты scp, что требуют дополнительных мер содержания. Программа изображена на Рисунке 6.

```

1     scp_objects = [
2         {"scp": "SCP-096", "class": "Euclid"},
3         {"scp": "SCP-173", "class": "Euclid"},
4         {"scp": "SCP-055", "class": "Keter"},
5         {"scp": "SCP-999", "class": "Safe"},
6         {"scp": "SCP-3001", "class": "Keter"}
7     ]
8
9     filtered = list(filter(lambda o: o['class'] != 'Safe', scp_objects))
10    print(filtered)

```

Рисунок 6 - Код для задания 6

Пояснение работы программы:

- 1) Получаем список словарей из задания
- 2) Создаём переменную `filtered` и придаём ей значения через функции `list()`, `filter()` и лямбда-выражение
- 3) В `filtered` попадают значения списка `scp_objects` если класс объекта не равен «Safe»
- 4) Выводим результат командой `print()`

1.7 Задание 7

В данном задании сделан код, сортирует инциденты по количеству задействованного персонала и показывает 3 наиболее ресурсоёмких из них. Программа изображена на Рисунке 7.

```

1     incidents = [
2         {"id": 101, "staff": 4},
3         {"id": 102, "staff": 12},
4         {"id": 103, "staff": 7},
5         {"id": 104, "staff": 20}
6     ]
7
8     sorted_i = sorted(incidents, key=lambda x: x['staff'], reverse=True)
9     top_3 = sorted_i[:3]
10    print(sorted_i)
11    print(top_3)

```

Рисунок 7 - Код для задания 7

Пояснение работы программы:

- 1) Получаем список словарей из задания
- 2) Создаём переменную `sorted_i` и придаём ей значение через функции `sorted()` и лямбда-выражение. Сортируем по значениям «staff», параметр `reverse=True` сортирует по убыванию
- 3) Создаём переменную `top_3` и она получает первые 3 значения из переменной `sorted_i`
- 4) Выводим результаты командой `print()`

1.8 Задание 8

В данном задании сделан код, который создаёт список словарей нового вида используя имя и уровень критичности. Программа изображена на Рисунке 8.

```

1     protocols = [
2         ("Lockdown", 5),
3         ("Evacuation", 4),
4         ("Data Wipe", 3),
5         ("Routine Scan", 1)
6     ]
7
8     newview = list(map(lambda x: f'Protocol {x[0]} - Criticality {x[1]}', protocols))
9     print(newview)

```

Рисунок 8 - Код для задания 8

Пояснение работы программы:

- 1) Получаем список словарей из задания
- 2) Создаём переменную newview через функции list(), map() и лямбда-выражение
- 3) Придаём ей значения строк в новом виде
- 4) Выводим результат командой print()

1.9 Задание 9

В данном задании сделан код, который выводит только те смены, что длятся не менее 8 часов и не более 12 часов. Программа изображена на Рисунке 9.

```

1     shifts = [6, 12, 8, 24, 10, 4]
2     filtered = list(filter(lambda x: 8 <= x <= 12, shifts))
3     print(filtered)

```

Рисунок 9 - Код для задания 9

Пояснение работы программы:

- 1) Получаем список длительности смен из задания
- 2) Создаём переменную filtered через функции list(), filter() и лямбда-выражение. Она получает значения только если смена находится в диапазоне от 8 до 12
- 3) Выводим результат командой print()

1.10 Задание 10

В данном задании сделан код, который находит наивысший результат из списка словарей работников и выводит в консоль его имя и результат. Программа изображена на Рисунке 10.

```
1     evaluations = [  
2         {"name": "Agent Cole", "score": 78},  
3         {"name": "Dr. Weiss", "score": 92},  
4         {"name": "Technician Moore", "score": 61},  
5         {"name": "Researcher Lin", "score": 88}  
6     ]  
7  
8     top = max(evaluations, key=lambda x: x["score"])  
9     print(f'Best employee - {top['name']}, Mental score - {top['score']}')
```

Рисунок 10 - Код для задания 10

Пояснение работы программы:

- 1) Получаем список словарей из задания
- 2) Создаём переменную top и придаём ей значения через функции max() и лямбда-выражения

- 3) Лямбда-выражение указывает искать лучшего сотрудника по ключу «score»
- 4) Выводим результат в консоль командой print()

Программа точно находит сотрудника с наивысшей психологической оценкой.