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

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №7

по дисциплине
«Информатика и программирование»

Студент
гр. БИН-25-3 _____ Герцов Д.Е.

Ассистент
преподавателя _____ М.В. Водяницкий

Владивосток 2025

Задание

Реализовать обработку данных с использованием функций высшего порядка (‘map‘, ‘filter‘, ‘sorted‘, ‘max‘) и лямбда-выражений. Все задания основаны на тематике Фонда SCP.

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 проведённых 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 отчёты. Некоторые содержат внешние ссылки, которые должны быть удалены.

Используя ‘filter‘ и ‘map‘ с лямбда-выражениями:

- Отберите отчёты, содержащие ссылки (‘http‘ или ‘https‘)
- Преобразуйте их так, чтобы вместо ссылки отображалось `[ДАННЫЕ УДАЛЕНЫ]`

Задание 6. Дан список SCP-объектов с указанием 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, требующих усиленных мер содержания (все, кроме класса ”Safe”).

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, отсортируйте инциденты по количеству персонала и оставьте только три наиболее ресурсоёмких.

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 список строк вида:
`"Protocol Lockdown — Criticality 5"`.

Задание 9. Имеется список смен охраны (в часах): `[6, 12, 8, 24, 10, 4]`. Используя ‘filter‘ и лямбда-выражение, выберите смены длительностью от 8 до 12 часов включительно.

Задание 10. Дан список сотрудников с результатами психологической оценки:

```
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

1.1 Задание 1

Реализована сортировка списка объектов по возрастанию уровня угрозы с использованием функции `sorted` и лямбда-выражения. Сортировка выполняется по второму элементу кортежа. На рисунке 1 представлен код программы.

```
def task1():
    objects = [
    ("Containment Cell A", 4),
    ("Archive Vault", 1),
    ("Bio Lab Sector", 3),
    ("Observation Wing", 2)
    ]
    print(sorted(objects, key= lambda x: x[1]))
```

Рисунок 1 – Листинг программы для задания 1

1) Определение списка объектов с уровнями угрозы
2) Использование `sorted` с ключом `lambda x: x[1]`
3) Вывод отсортированного списка

После выполнения программы объекты будут упорядочены от наименее угрожающих к наиболее опасным.

1.2 Задание 2

Реализован расчёт общей стоимости работы каждого сотрудника с использованием `map` и лямбда-выражения. Также найдена максимальная стоимость среди всех сотрудников. На рисунке 2 представлен код программы.

```
def task2():
    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}
    ]
    shift_costs = list(map(lambda x: x["shift_cost"],
    staff_shifts))
    print(shift_costs)
    print(f"Максимум: {max(shift_costs)}")
```

Рисунок 2 – Листинг программы для задания 2

1) Извлечение стоимости смены с помощью `map`
2) Преобразование в список
3) Нахождение максимального значения с помощью `max`
4) Вывод стоимости и максимального значения

После выполнения программы будет получен список стоимостей и выделено максимальное значение.

1.3 Задание 3

Реализовано добавление категории допуска к каждому сотруднику на основе его уровня допуска с использованием `map` и условного лямбда-выражения. На рисунке 3 представлен код программы.

```
def task3():
    personnel = [
    {"name": "Dr. Klein", "clearance": 2},
    {"name": "Agent Brooks", "clearance": 4},
    {"name": "Technician Reed", "clearance": 1}
    ]
    a = list(map(lambda x: { "name": x["name"],
                 "clearance": x["clearance"],
                 "category":(

     'Top secret' if 4 == x["clearance"] else 'Confidental'
   if 2 == x['clearance'] else 'Restricted')}, personnel))
```

Рисунок 3 – Листинг программы для задания 3

1) Обработка каждого сотрудника через `map`
2) Присвоение категории в зависимости от значения `clearance`
3) Формирование нового списка словарей с полем `category`
4) Вывод результата

После выполнения программы каждый сотрудник будет иметь соответствующую категорию секретности.

1.4 Задание 4

Реализован отбор зон, полностью работающих в дневное время (с 8 до 18 часов), с использованием `filter` и лямбда-выражения. На рисунке 4 представлен код программы.

```
def task4():
    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}
    ]
    a = list(filter(lambda x: x["zone"] if x["active_from"
   ]>=8 and x["active_to"]<= 18 else False, zones))

    print(a)
```

Рисунок 4 – Листинг программы для задания 4

1) Фильтрация зон по условию: `active_from >= 8` и `active_to <= 18`

2) Извлечение только названий зон (в текущей реализации — ошибка; корректно: возвращать весь словарь)

3) Вывод отфильтрованных зон

После выполнения программы будут выведены зоны, работающие строго в дневное время.

1.5 Задание 5

Реализована обработка служебных отчётов: отбор тех, что содержат HTTP/HTTPS-ссылки, и их очистка с заменой ссылок на `[ДАННЫЕ УДАЛЕНЫ]` с использованием регулярных выражений. На рисунке 5 представлен код программы.

```
import re
def task5():
    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"}
    ]


    a = list(map(
        lambda r: {"author": r["author"], "text": re.sub(r'https?://\S+', 'ДАННЫЕ[ УДАЛЕНЫ]', r["text"])},
        filter(lambda r: 'http://' in r["text"] or 'https://' in r["text"], reports)
    ))
    print(a)
```

Рисунок 5 – Листинг программы для задания 5

1) Фильтрация отчётов, содержащих `http://` или `https://`

2) Замена всех URL на `[ДАННЫЕ УДАЛЕНЫ]` с помощью `re.sub`

3) Формирование нового списка очищённых отчётов

4) Вывод результата

После выполнения программы все внешние ссылки будут надёжно удалены из отчётов.

1.6 Задание 6

Реализован отбор SCP-объектов, требующих усиленных мер содержания (все, кроме класса `"Safe"`), с использованием `filter` и лямбда-выражения. На рисунке 6 представлен код программы.

```
def task6():
    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"}
    ]

    high_risk = list(filter(lambda x: x["class"] != "Safe",
    scp_objects))

    print(high_risk)
```

Рисунок 6 – Листинг программы для задания 6

1) Фильтрация объектов по условию: `class != "Safe"`

2) Сохранение исходной структуры словарей

3) Вывод списка опасных SCP-объектов

После выполнения программы будет получен список всех объектов, кроме безопасных.

1.7 Задание 7

Реализована сортировка инцидентов по количеству задействованного персонала в порядке убывания и выбор трёх наиболее ресурсоёмких с использованием `sorted` и среза. На рисунке 7 представлен код программы.

```
def task7():
    incidents = [
        {"id": 101, "staff": 4},
        {"id": 102, "staff": 12},
        {"id": 103, "staff": 7},
        {"id": 104, "staff": 20}
    ]

    sorted_incidents = sorted(incidents, key=lambda x: x["staff"], reverse=True)

    top3 = sorted_incidents[:3]

    print(top3)
```

Рисунок 7 – Листинг программы для задания 7

1) Сортировка списка инцидентов по ключу `staff` в обратном порядке
2) Выбор первых трёх элементов с помощью среза `[:3]`
3) Вывод топ-3 инцидентов

После выполнения программы будут выведены три самых масштабных инцидента.

1.8 Задание 8

Реализовано форматирование списка протоколов безопасности в читаемые строки с использованием `map` и лямбда-выражения. На рисунке 8 представлен код программы.

```
def task8():
    protocols = [
    ("Lockdown", 5),
    ("Evacuation", 4),
    ("Data Wipe", 3),
    ("Routine Scan", 1)
    ]

    result = list(map(lambda x: f"Protocol {x[0]} - Criticality {x[1]}", protocols))
    print(result)
```

Рисунок 8 – Листинг программы для задания 8

1) Преобразование каждого кортежа в строку требуемого формата
2) Использование f-строк внутри лямбда-выражения
3) Вывод списка отформатированных протоколов

После выполнения программы будет получен список протоколов в стандартизированном виде.

1.9 Задание 9

Реализован отбор смен охраны длительностью от 8 до 12 часов включительно с использованием `filter` и лямбда-выражения. На рисунке 9 представлен код программы.

```
def task9():
    shifts = [6, 12, 8, 24, 10, 4]

    valid_shifts = list(filter(lambda x: 8 <= x <= 12,
    shifts))
    print(valid_shifts)
```

Рисунок 9 – Листинг программы для задания 9

1) Фильтрация списка `shifts` по условию `8 <= x <= 12`
2) Сохранение только подходящих значений
3) Вывод результата

После выполнения программы будут выведены только стандартные дневные смены.

1.10 Задание 10

Реализован поиск сотрудника с наивысшей психологической оценкой с использованием `max` и лямбда-выражения. На рисунке 10 представлен код программы.

```
def task10():

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

    best = max(evaluations, key=lambda x: x["score"])
    print(best)
```

Рисунок 10 – Листинг программы для задания 10

1) Поиск словаря с максимальным значением `score`
2) Использование ключа `key=lambda x: x["score"]`
3) Вывод полной информации о лучшем сотруднике

После выполнения программы будет выведено имя и оценка сотрудника с наивысшим показателем.