**Тема:** *Программные измерительные мониторы*

**Цель задания:**

1. Понять назначение программных мониторов для измерения характеристик кода.
2. Научиться применять простые инструменты мониторинга (профилировщики, системные утилиты).
3. Анализировать результаты замеров (время выполнения, использование памяти).

**Теоретическая часть (краткие ответы)**

1. **Определение:**  
   Что такое *программный измерительный монитор*? Приведите три примера инструментов.
2. **Задачи мониторинга:**  
   Перечислите 3 ключевые характеристики ПО, которые можно измерить с помощью мониторов.
3. **Этапы работы:**  
   Опишите последовательность действий: запуск программы с монитором → сбор данных → интерпретация результатов.

**Практическая часть (работа с инструментами)**

**Задача:** Сравнить эффективность двух алгоритмов поиска элемента в массиве: линейного и бинарного.

**Код для анализа (Python): файл test6.py и test7.py**

**Задание 1: Измерение времени выполнения**

1. Сгенерируйте **отсортированный массив** из 10 000 000 случайных чисел.
2. Замерьте время выполнения для поиска **крайнего элемента** (последнего в массиве) с помощью:
3. *# Для Linux/MacOS:*
4. time python linear\_search.py
5. time python binary\_search.py
6. *# Для Windows:*
7. Measure-Command {python test6.py} *# Линейный*
8. Measure-Command {python test7.py}*# Бинарный*
9. Результаты занесите в таблицу:

| **Алгоритм** | **Реальное время(ms)** | **Использование памяти** |
| --- | --- | --- |
| Линейный поиск |  |  |
| Бинарный поиск |  |  |

**Задание 2: Измерение использования памяти**

1. Установите модуль memory\_profiler:

pip install memory\_profiler

1. Добавьте в код декораторы для замера памяти:

from memory\_profiler import profile

@profile

def linear\_search(arr, target): ...

@profile

def binary\_search(arr, target): ...

1. Запустите каждый скрипт и запишите **пиковое использование памяти** (MiB).

**Задание 3: Анализ результатов**

1. Какой алгоритм работает **быстрее** на больших данных? Объясните, почему.
2. Какой алгоритм использует **меньше памяти**? Почему?
3. Какие **ограничения** имеют использованные инструменты (time, memory\_profiler)?

**Форма отчета(два дополненных файла .py и .doc):**

1. **Теоретическая часть:** Ответы на вопросы.
2. **Таблицы с результатами** замеров (время, память).
3. **Анализ:** Ответы на вопросы из Задания 3.
4. **Выводы:**
   * Для каких задач подходит каждый алгоритм?
   * Когда применение измерительных мониторов критически важно?