# **Отчёт** по Lab-5 Солоницкий Максим Алексеевич М3234.

# **Конфигурация системы:** Янаписал специальный скрипт, который выводит все нужные данные в удобном формате: **Изображение выглядит как текст, снимок экрана, Шрифт Автоматически созданное описание**

# **Общий объем виртуальной памяти - 1968 MiB**

# **Объем раздела подкачки - 2679 MiB**

# **Размер страницы виртуальной памяти - 4096 MiB**

# **Объем свободной физической системы в ненагруженной системе - 850 MiB**

# **Объем свободного пространства в разделе подкачки в ненагруженной системе - 2388 MiB**

# **Эксперимент 1.1:**

# **Описание эксперимента:**

1. Написан и запущен скрипт mem.bash, который бесконечно добавляет элементы в массив.
2. Фиксировались изменения параметров памяти и поведения системы.

**Результаты:**   
  
**Последняя запись в системном журнале:**  
  

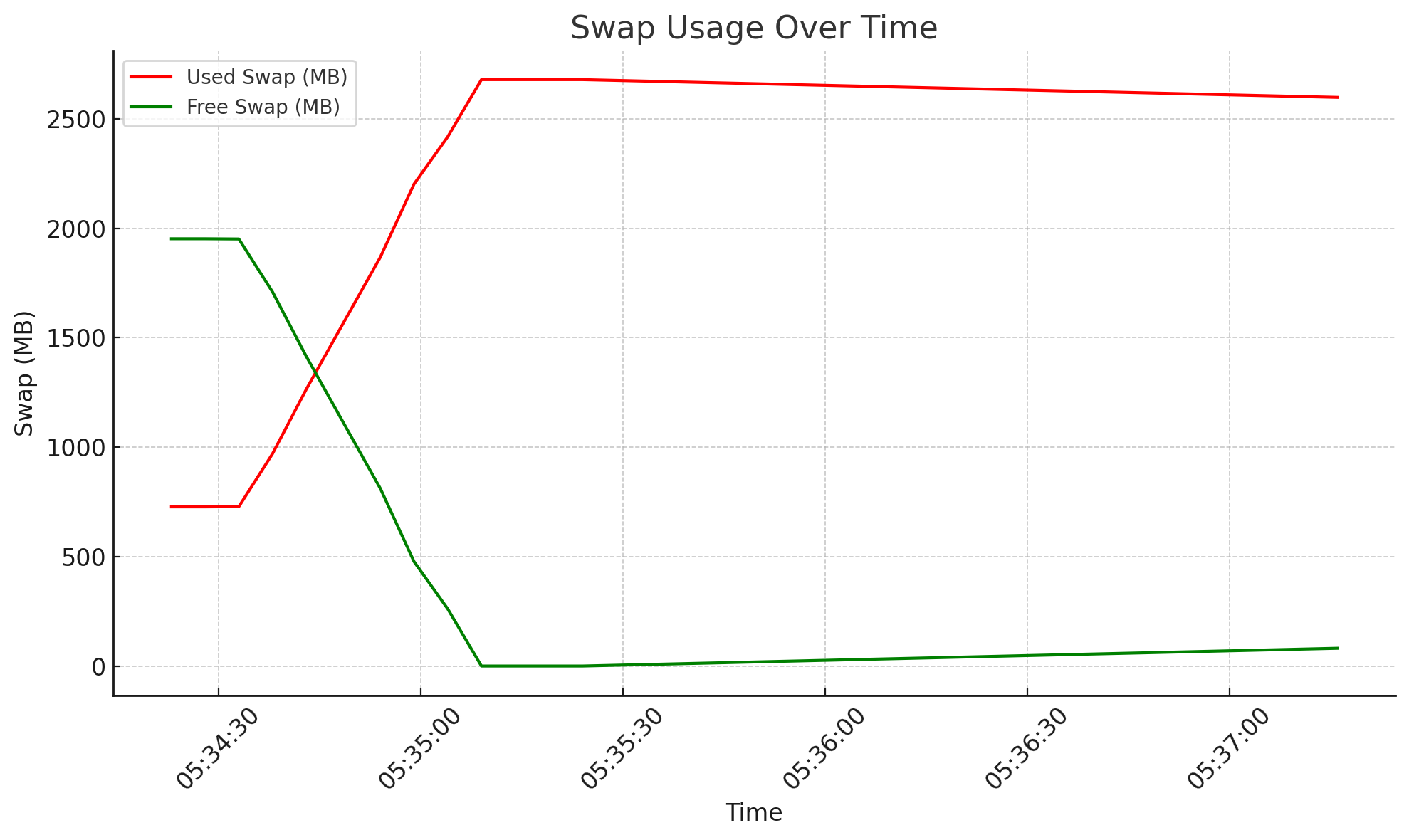

**Последняя запись в report.log:**  

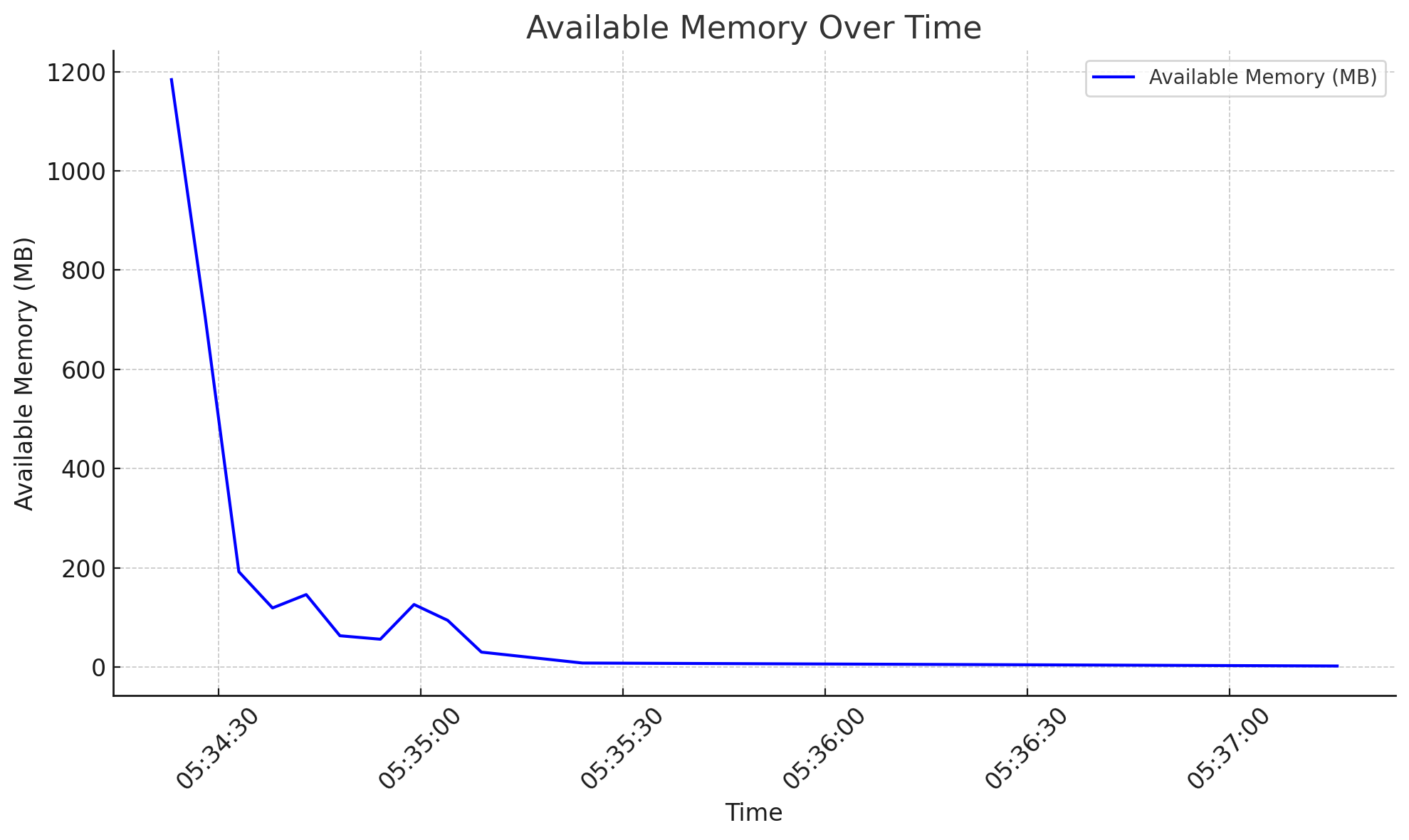

**График 1 (памяти):**

Этот график демонстрирует, как система постепенно исчерпывает RAM и перераспределяет ресурсы между кешем и активными процессами.

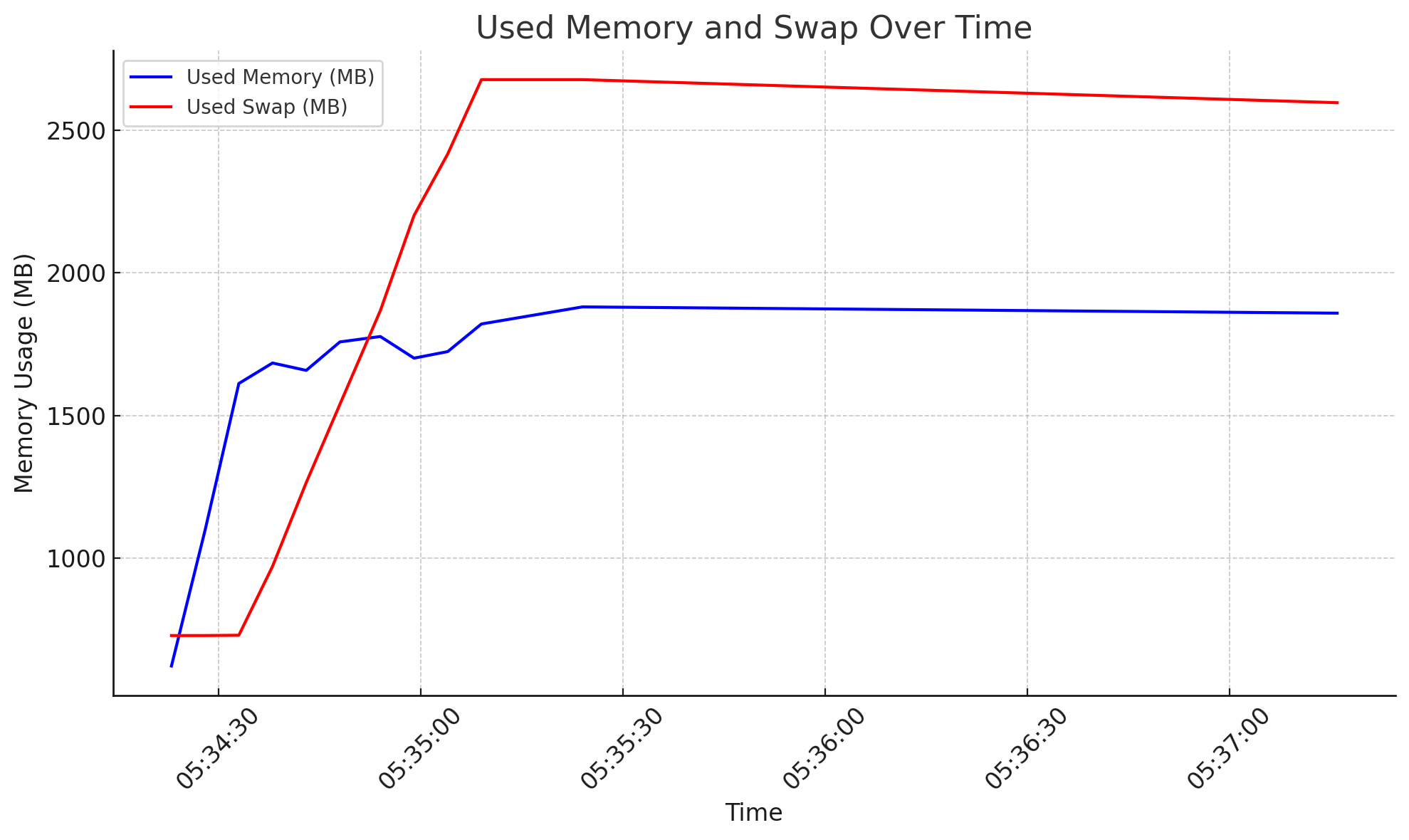
# 

**График 2 (swap):**

На этом графике видна динамика использования swap.  
Swap помогает системе продолжить работу после исчерпания оперативной памяти, но его использование приводит к значительному замедлению работы.  
  
**График 3 (доступная память):**

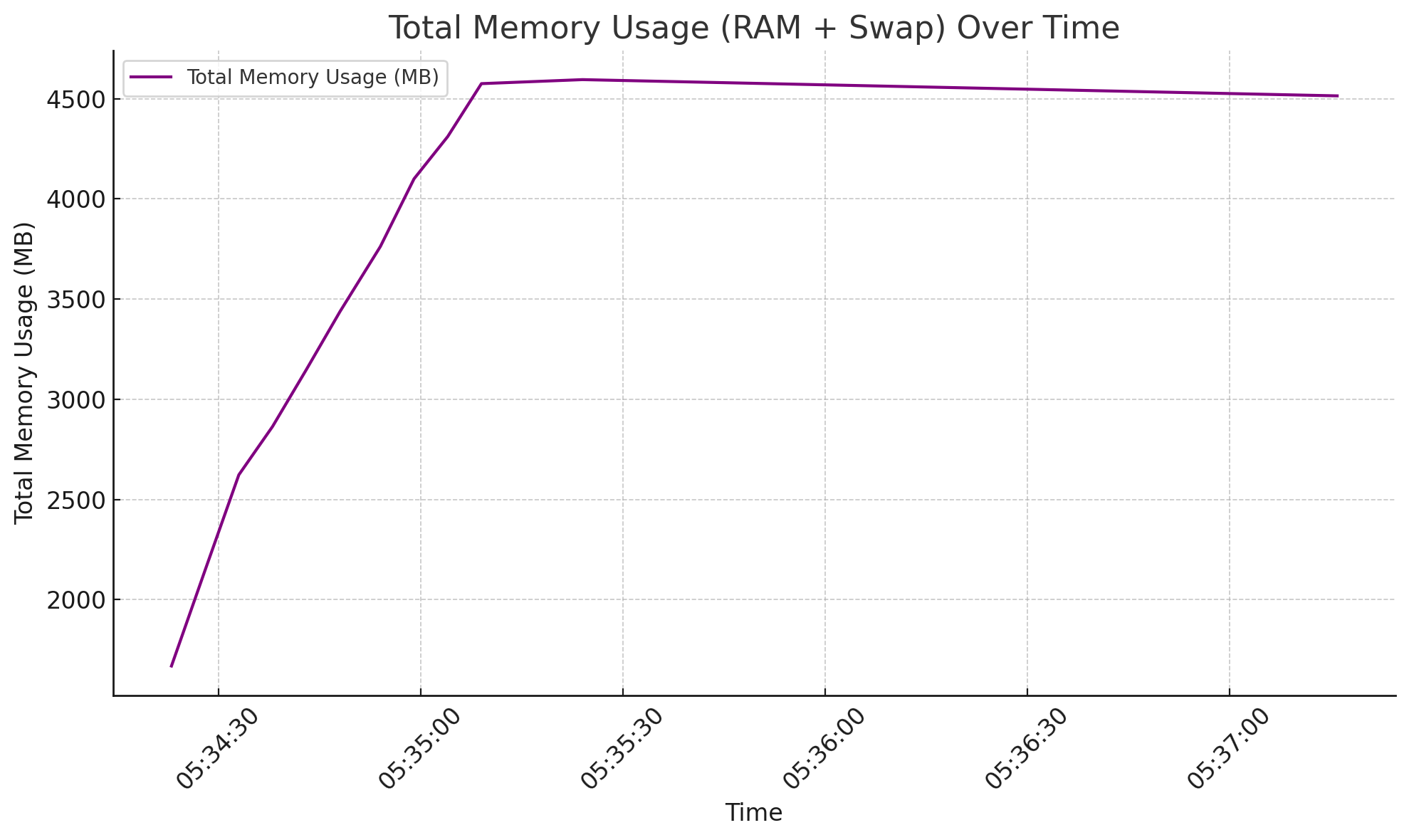
Этот график позволяет наглядно понять момент, когда система переходит в режим интенсивного использования swap.

**График 4 (соотношение использованной памяти и swap):**

Этот график иллюстрирует переход между уровнями памяти и показывает, как swap берет на себя основную нагрузку после исчерпания оперативной памяти. ****

**График 5 (полное использование памяти):**

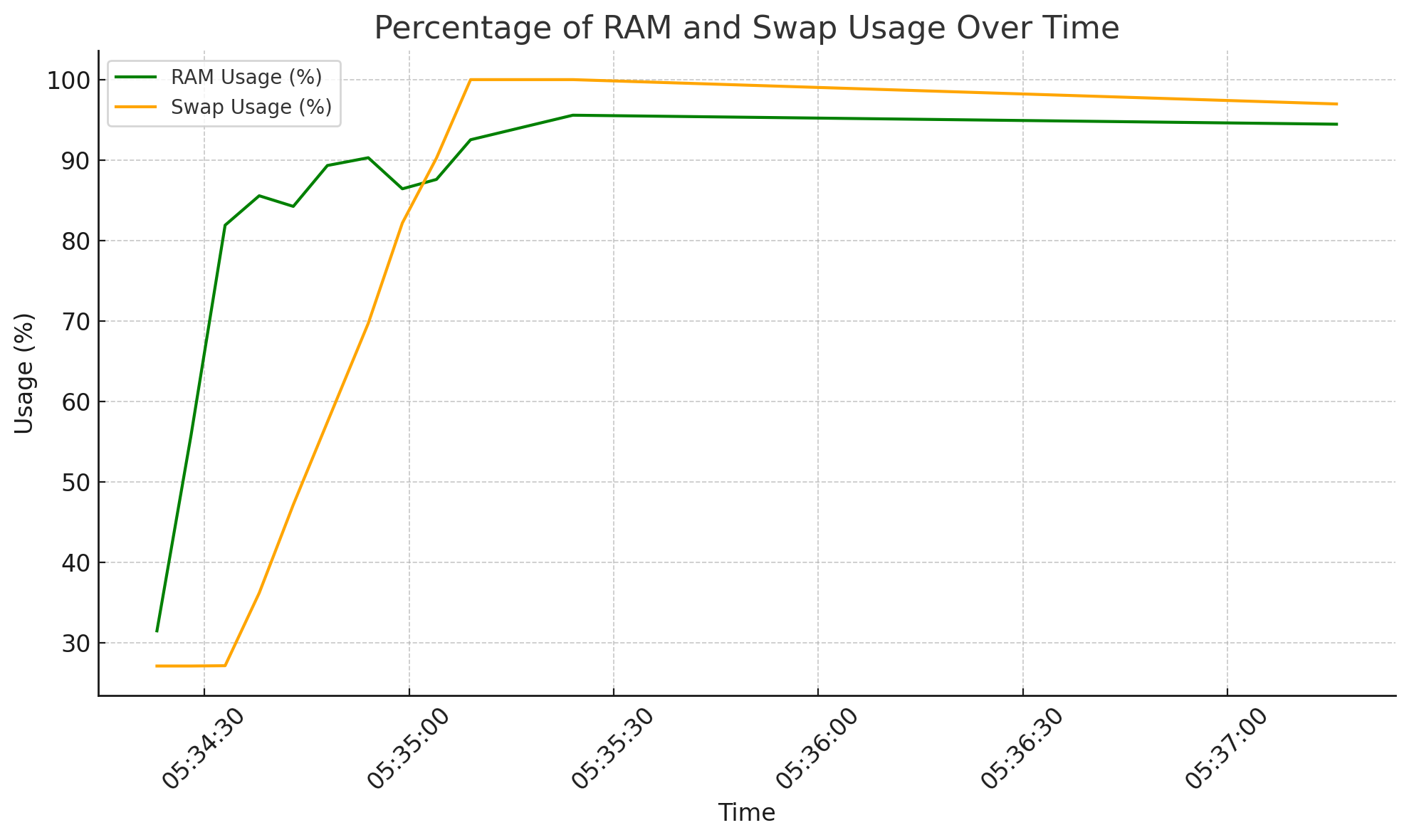
Этот график дает целостное представление о том, как растет общая потребность процесса в памяти.

****

**График 6 (процент использования ресурсов):**

Процентное использование памяти и swap наглядно отражает:

* Полное заполнение оперативной памяти, достигающее 100%.
* Быстрое заполнение swap после исчерпания RAM.

Система эффективно использует ресурсы памяти, но при полном заполнении swap теряет возможность продолжать работу. **  
  
Вывод по эксперименту 1.1:**

Динамика работы памяти:

* Система использует RAM до 100%, после чего переходит на swap.
* Swap компенсирует нехватку оперативной памяти, но его использование увеличивает нагрузку на диск, замедляя выполнение процессов.

Критические моменты:

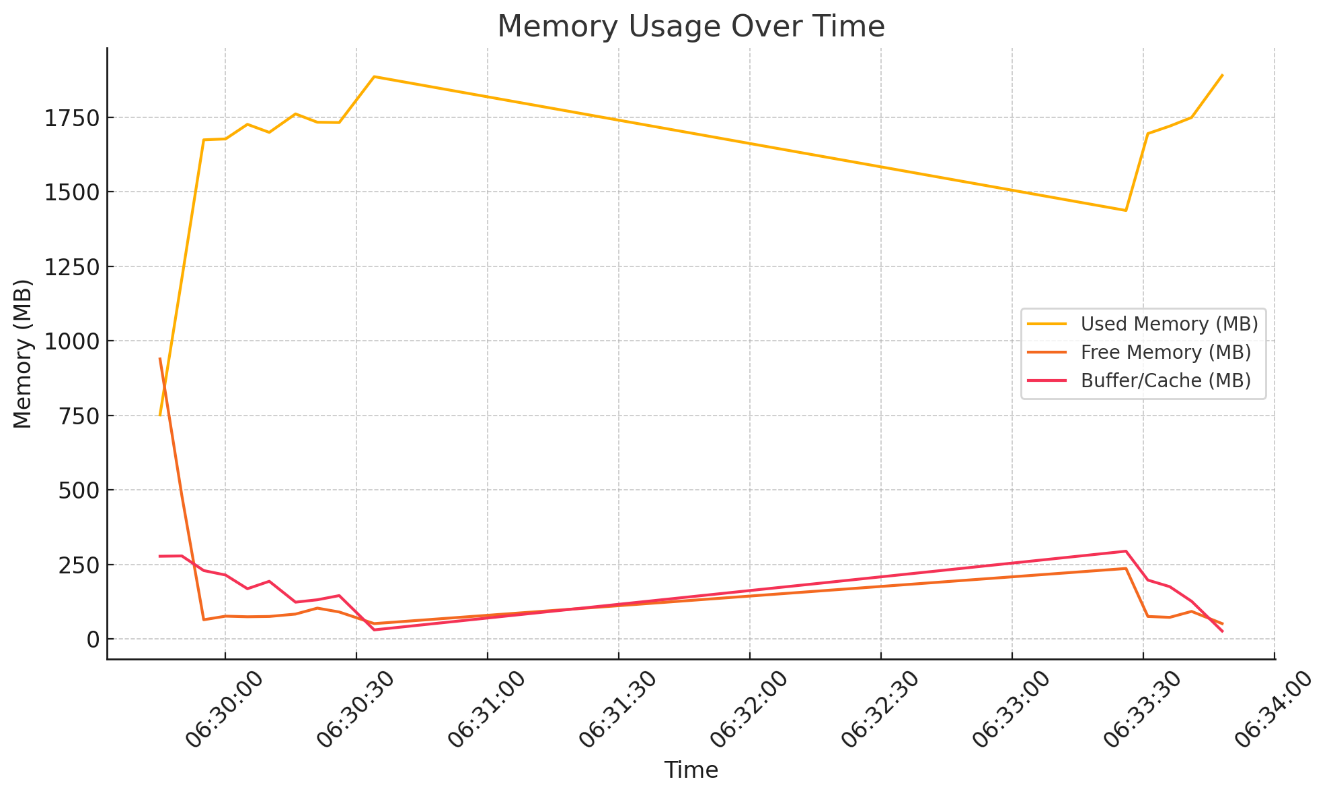
* Заполнение RAM до 100% (~1800 MB).
* Заполнение swap до 100% (~2679 MB).
* Последующая аварийная остановка процесса.

Общее поведение системы:

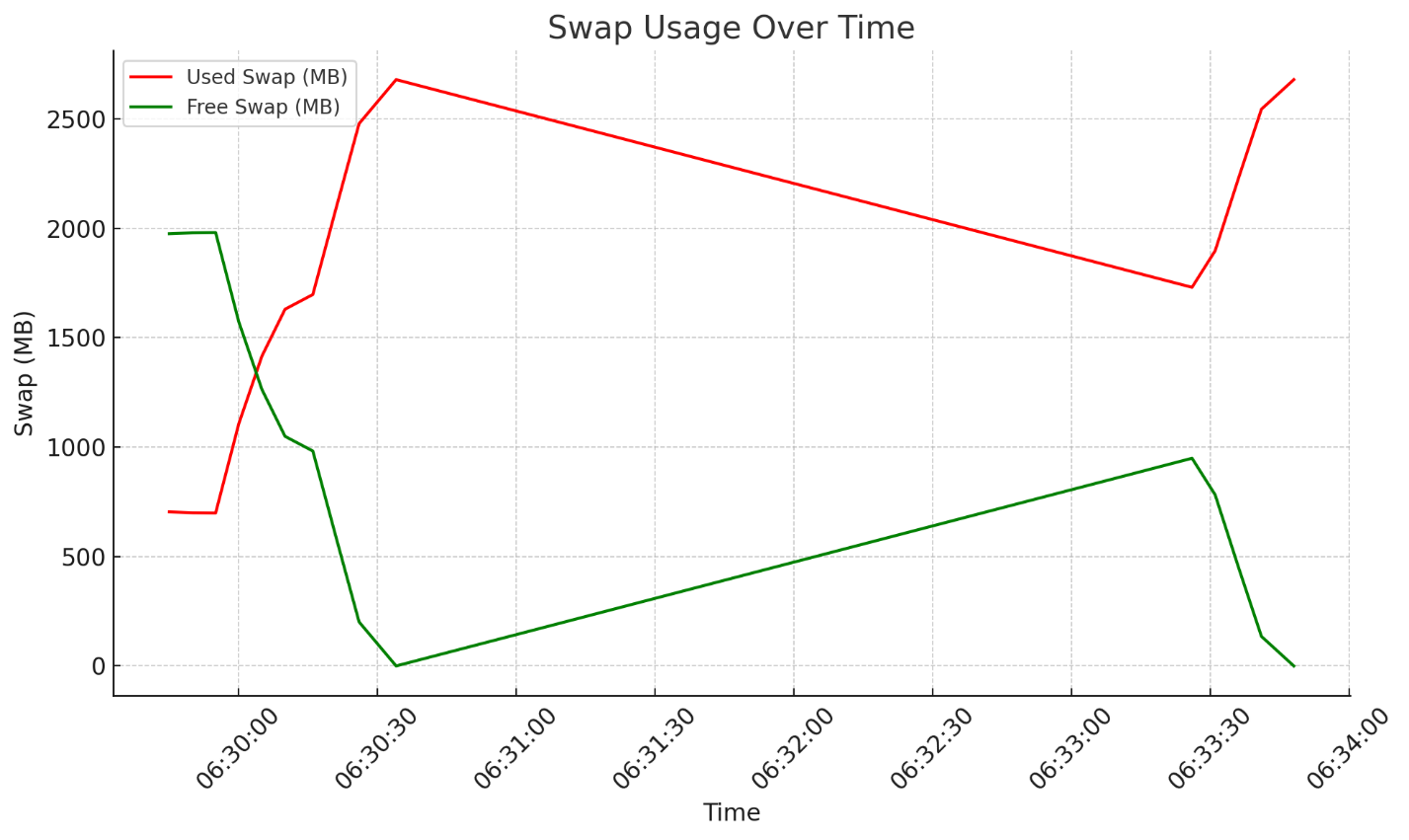
* После полного заполнения оперативной памяти и swap процесс становится недоступен для выполнения, что фиксируется в логах.

# **Эксперимент 1.2:**

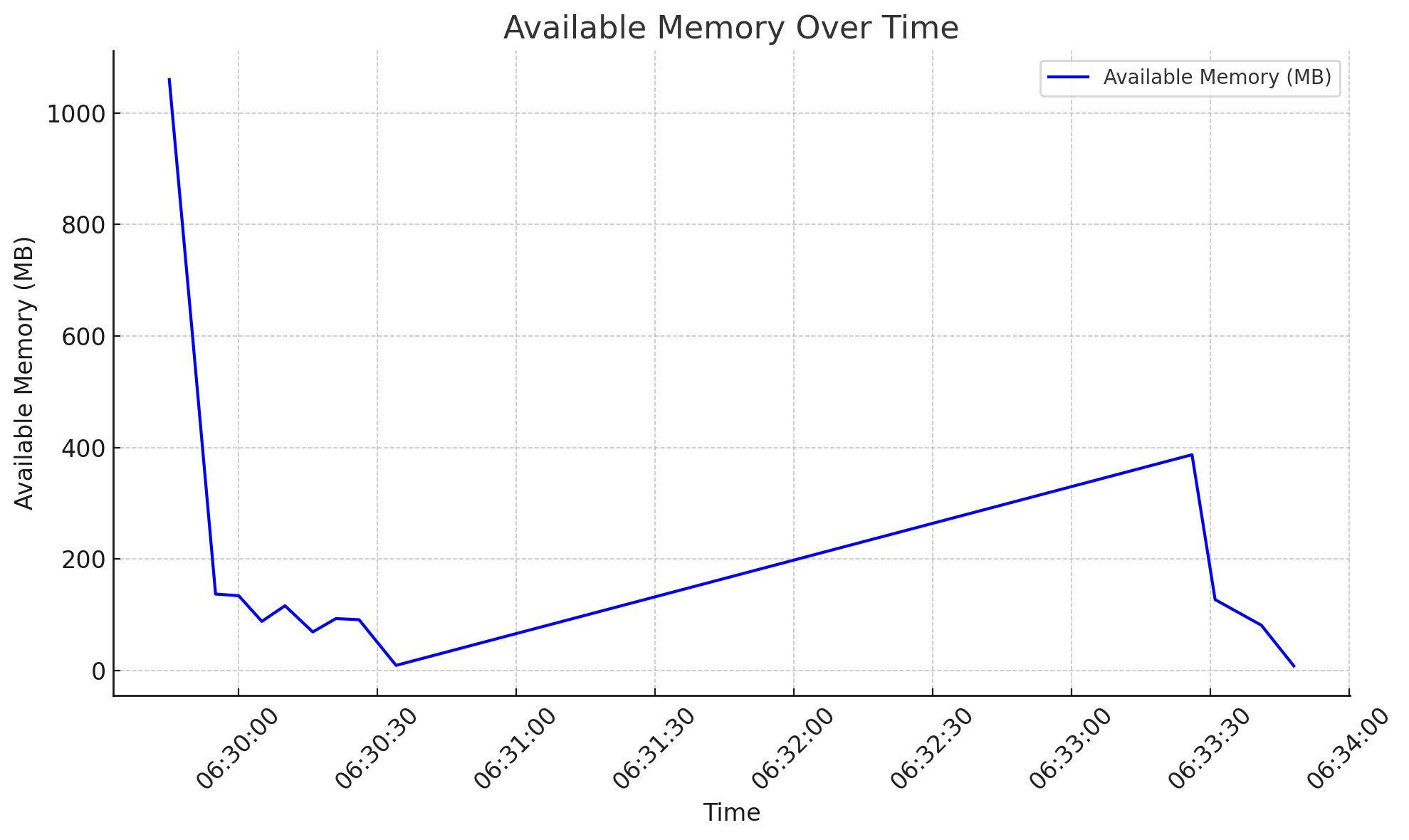
**График 1 (памяти):**

****

**График 2 (swap):**

****

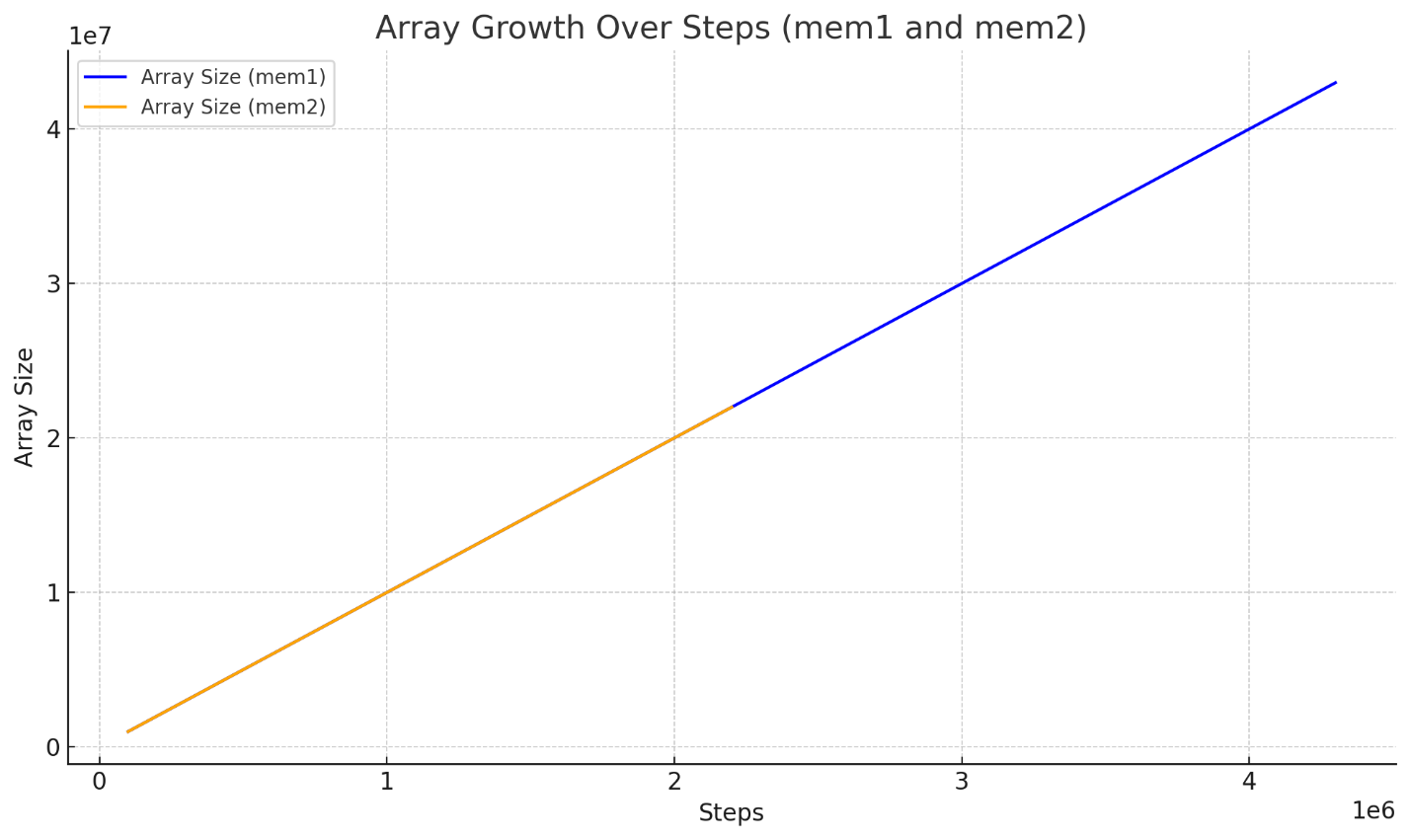
**График 3 (доступная память):**

****

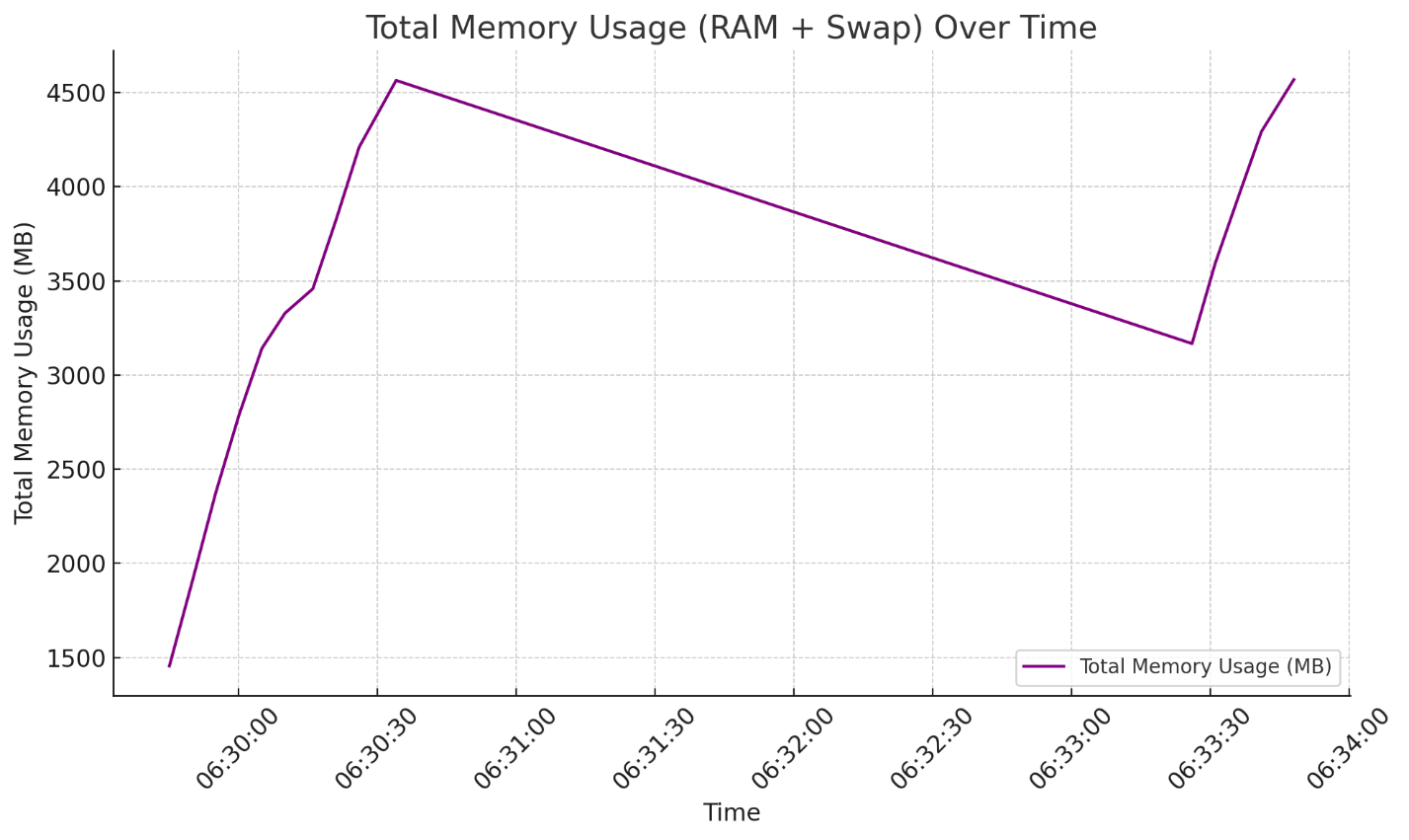
**График 4 (соотношение использованной памяти и swap):**

Оба процесса (mem1 и mem2) наращивают массивы равномерно.

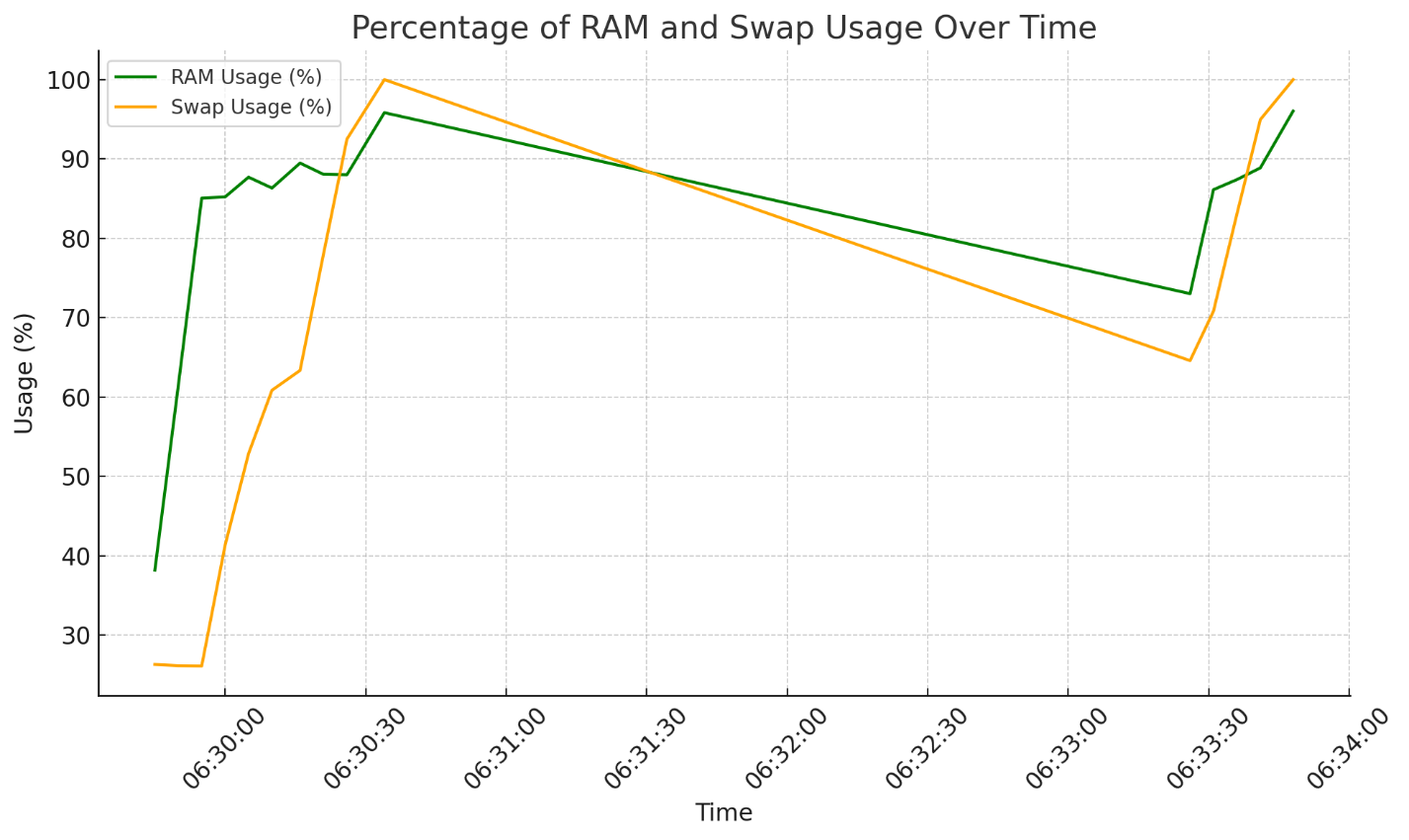
Процесс mem1 продолжает расти дольше, так как mem2 завершает работу раньше, вероятно, из-за недостатка памяти.

****

**График 5 (полное использование памяти):**

****

**График 6 (процент использования ресурсов):**

****

**Вывод по эксперименту 1.2:**

1. Динамика памяти:
   * Оперативная память полностью используется обоими процессами, что приводит к активному использованию swap.
   * Полное заполнение swap вызывает критическую перегрузку системы, что приводит к завершению работы процессов.
2. Поведение процессов:
   * mem1 продолжает работу дольше, пока mem2 завершается из-за исчерпания ресурсов.
   * Оба процесса активно конкурируют за память, что ускоряет использование swap.

**Последняя запись в системном журнале:**  
  


**Значение в последней строке файла report2\_1.log - 4300000**

**Значение в последней строке файла report2.log – 2200000**

**Эксперимент 2:**

Возьмем N = 3900000 / 10 и K = 10 и интервалом 1 секунду, скрипт работает аварийного завершения нет. (аналогично с интервалами до 0.1 секунды).

При K = 30, некоторые процессы завершились аварийно (все процессы с 19го) из - за того, что не всем им хватило памяти.

С помощью бинарного поиска найдем граничное значение N, при котором все будет работать. Результатом вычисления будет N = 2887500, при N большем происходит аварийное завершение процессов (и, вообще, крашится виртуалка).

**Выводы:**

1. **Граничное значение:**
   * Для стабильной работы 30 процессов с интервалом запуска 1 секунда максимальное значение N, обеспечивающее успешное завершение всех процессов: **N = 2887500**.
2. **Поведение системы:**
   * При увеличении N свыше граничного значения наблюдается:
     + Исчерпание оперативной памяти и swap.
     + Массовое аварийное завершение процессов.
     + Потенциальная нестабильность системы (краш виртуальной машины).

Этот эксперимент показывает важность грамотного управления ресурсами в многозадачных средах и демонстрирует, как баланс между числом процессов, объемом памяти и интервалом запуска влияет на стабильность системы.