

Betriebssysteme - Praktikum 1

Aufgabe 1

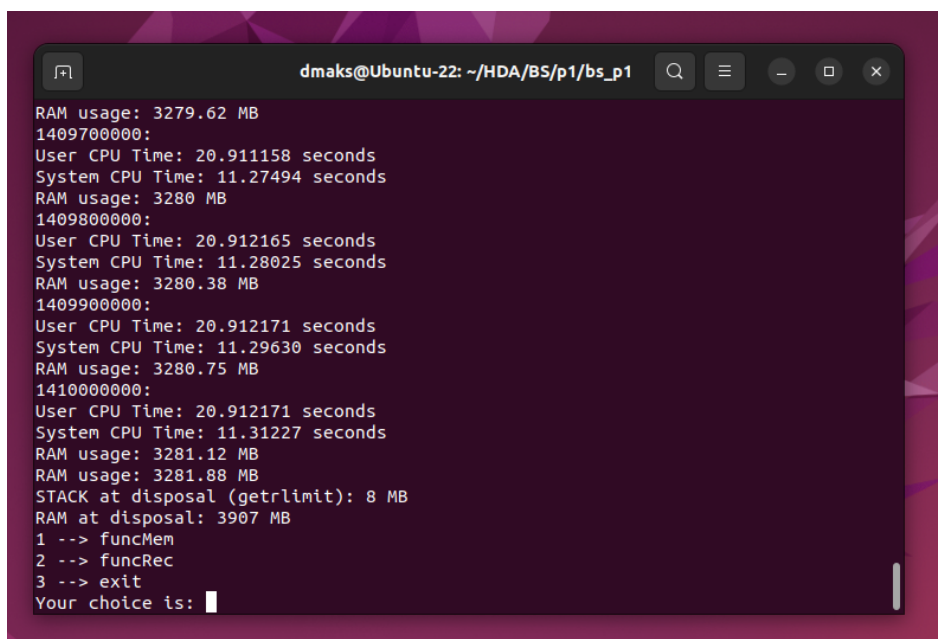
a) Dokumentieren Sie die maximal nutzbare Größe an Hauptspeicher sowie die maximal nutzbare Stackgröße, getrennt als experimentell ermittelte bzw. aus dem Betriebssystem über `getrlimit` ausgelesene Größen

Die maximal nutzbare Größe an Hauptspeicher (RAM) haben wir innerhalb der `funcMem` experimentell festgestellt und zwar in dem wir innerhalb der `for`-schleife, in bestimmte frequenzen (z.B. 1000 Iterationen) die Funktion `ramUsageProvider` aufgerufen haben. Danach wird diese Funktion über `getrusage` die aktuelle RAM-Usage zurückliefern. Um maximale Größe RAMs festzustellen, wählen wir eine sehr große Array, dadurch sind wir in der Lage RAM-Usage zu folgen und festzustellen was die maximal nutzbare Größe an RAM ungefähr wäre, weil in einem Moment RAM-Usage zum Zeitpunkt kommen wird, wenn es entweder crasht (Siehe Bild 1) oder schafft das (Siehe Bild 2) Programm auszuführen. Auf jedem Fall, durch konstante ausgabe des aktuellen Wertes das das RAM nutzt, stellen wir fest, was die maximal nutzbare Größe RAMs ungefähr wäre.



```
bs_p1
1364700000:
User CPU Time: 19.909458 seconds
System CPU Time: 10.613571 seconds
RAM usage: 3242.75 MB
1364800000:
User CPU Time: 19.925949 seconds
System CPU Time: 11.31701 seconds
RAM usage: 3243.12 MB
1364900000:
User CPU Time: 19.925949 seconds
15:46:27: /home/dmaks/HDA/BS/p1/build-bs_p1-Desktop_Qt_6_6_0_GCC_64bit-Debug/bs_p1 crashed.
```

Bild 1, Absturz Programmes (`funcMem`) in Qt-Creator



```
dmaks@Ubuntu-22: ~/HDA/BS/p1/bs_p1
RAM usage: 3279.62 MB
1409700000:
User CPU Time: 20.911158 seconds
System CPU Time: 11.27494 seconds
RAM usage: 3280 MB
1409800000:
User CPU Time: 20.912165 seconds
System CPU Time: 11.28025 seconds
RAM usage: 3280.38 MB
1409900000:
User CPU Time: 20.912171 seconds
System CPU Time: 11.29630 seconds
RAM usage: 3280.75 MB
1410000000:
User CPU Time: 20.912171 seconds
System CPU Time: 11.31227 seconds
RAM usage: 3281.12 MB
RAM usage: 3281.88 MB
STACK at disposal (getrlimit): 8 MB
RAM at disposal: 3907 MB
1 --> funcMem
2 --> funcRec
3 --> exit
Your choice is: █
```

Bild 2, Programm liefert Ergebnisse (`funcMem`) in Terminal

Obwohl wir 4GB RAM als Ergebnis Experiments erwarten, kommen wir nicht dazu. Das liegt sehr wahrscheinlich daran, dass wegen der Belastung Programms, aktuelle RAM-Usage nicht richtig ausgegeben wird (Siehe Bild 3).

```
dmaks@Ubuntu-22: ~/HDA/BS/p1/bs_p1

System CPU Time: 47.641103 seconds
RAM usage: 3222.12 MB
1410056000:
User CPU Time: 29.606035 seconds
System CPU Time: 47.641103 seconds
RAM usage: 3222.12 MB
1410057000:
User CPU Time: 29.606084 seconds
System CPU Time: 47.641103 seconds
RAM usage: 3222.12 MB
1410058000:
User CPU Time: 29.606165 seconds
System CPU Time: 47.641103 seconds
RAM usage: 3222.12 MB
1410059000:
User CPU Time: 29.606215 seconds
System CPU Time: 47.641103 seconds
RAM usage: 3222.12 MB
1410060000:
User CPU Time: 29.606268 seconds
System CPU Time: 47.641103 seconds
RAM usage: 3222.12 MB
1410061000:
User CPU Time: 29.606319 seconds
System CPU Time: 47.641103 seconds
RAM usage: 3222.12 MB
1410062000:
User CPU Time: 29.606361 seconds
System CPU Time: 47.641103 seconds
RAM usage: 3222.12 MB
1410063000:
User CPU Time: 29.606403 seconds
System CPU Time: 47.641103 seconds
RAM usage: 3222.12 MB
1410064000:
User CPU Time: 29.606444 seconds
System CPU Time: 47.641103 seconds
RAM usage: 3222.12 MB
```

Bild 3, Wiederholung der Ergebnisse (funcMem)

Unsere Behauptung, dass es sich um einen Bug handelt, prüfen wir mithilfe von System Monitor in Ubuntu. Während der Ausführung unseres Programmes, wird fast 100% RAM benutzt werden (Siehe Bild 4). Und da wir 4GB RAM verwenden können, wissen wir dass Ausgabe Programmes nicht voll korrekt ist, obwohl unsere Methode mit dem wir Experiment durchführen richtig ist.

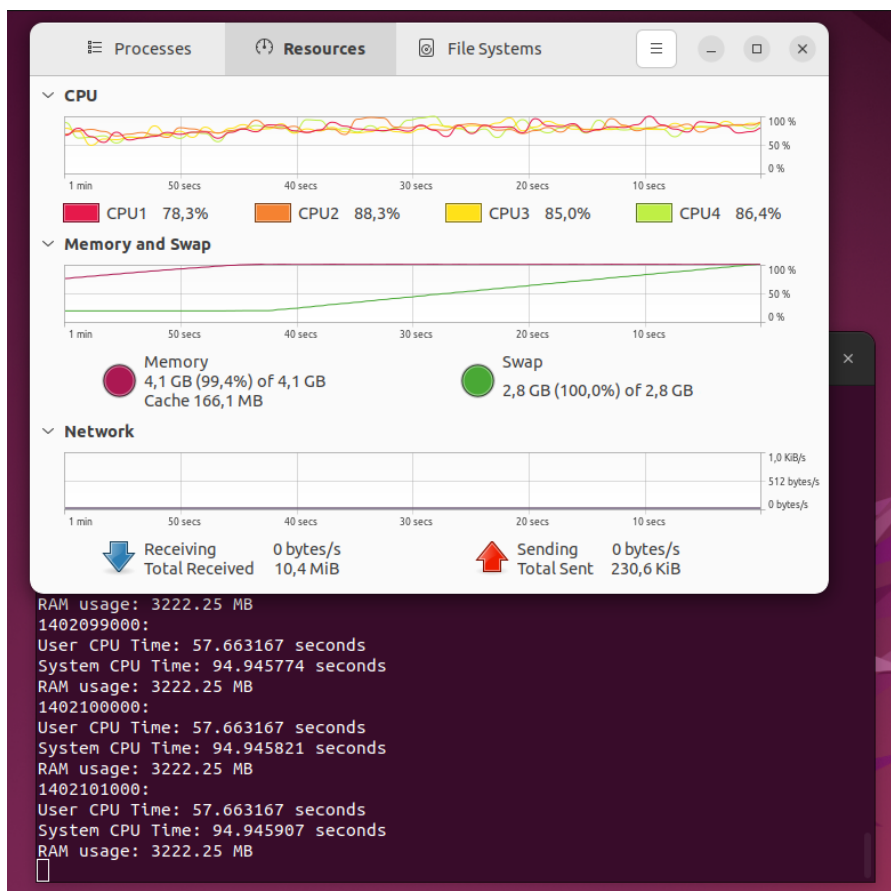


Bild 4, System Monitor während Ausführung Programmes (funcMem)

Um echte Größe des RAMs festzustellen, nutzen wir funktion `storedMemoryInfo`. Diese Funktion liefert immer, das Ergebnis das wir erwarten und zwar etwa 4GB (Siehe Bild 2, „RAM at disposal“).

Unser Experiment zeigt, dass wir 4GB RAM zur Verfügung haben.

Die maximal nutzbare Stackgröße haben wir experimentell mithilfe von function *funcRec* berechnet, in dem wir vor der Ausführung der Funktion, auf Frame Stacks einen Char „stackStart“ geschrieben haben, und einen Pointer auf Adresse dieses Chars gespeichert haben. Innerhalb der Funktion *funcRec* wird dann nach entsprechenden Anzahl Iterationen, die aktuelle Stack-Usage zurückgeliefert, mithilfe noch eines Chars „stackCurrent“ bzw. „stackEnd“, der auf Stack geschrieben wird. Danach wird die Differenz der Adressen der Char (in Bytes) in MB umgewandelt und als „estimated stack usage“ ausgegeben werden (Siehe Bilder 5 und 6).

```
Estimated current stack usage: 7.55308 MB
User CPU Time: 0.6927 seconds
System CPU Time: 0.12269 seconds
STACK at disposal (getrlimit): 8 MB
RAM at disposal: 3907 MB
Estimated stack usage: 7.62938 MB
1 --> funcMem
2 --> funcRec
3 --> exit
Your choice is:
```

Bild 5, Stack limit nicht überschritten (*funcRec*)

```
System CPU Time: 0.13730 seconds
Estimated current stack usage: 6.79014 MB
User CPU Time: 0.6271 seconds
System CPU Time: 0.13901 seconds
Estimated current stack usage: 6.86644 MB
User CPU Time: 0.6271 seconds
System CPU Time: 0.14105 seconds
Estimated current stack usage: 6.94273 MB
User CPU Time: 0.6271 seconds
System CPU Time: 0.14293 seconds
Estimated current stack usage: 7.01902 MB
User CPU Time: 0.6271 seconds
System CPU Time: 0.14512 seconds
Estimated current stack usage: 7.09532 MB
User CPU Time: 0.6271 seconds
System CPU Time: 0.14695 seconds
Estimated current stack usage: 7.17161 MB
User CPU Time: 0.6271 seconds
System CPU Time: 0.14871 seconds
Estimated current stack usage: 7.24791 MB
User CPU Time: 0.6271 seconds
System CPU Time: 0.15079 seconds
Estimated current stack usage: 7.3242 MB
User CPU Time: 0.6271 seconds
System CPU Time: 0.15254 seconds
Estimated current stack usage: 7.40049 MB
User CPU Time: 0.6271 seconds
System CPU Time: 0.15427 seconds
Estimated current stack usage: 7.47679 MB
User CPU Time: 0.6271 seconds
System CPU Time: 0.15624 seconds
Estimated current stack usage: 7.55308 MB
User CPU Time: 0.6271 seconds
System CPU Time: 0.15803 seconds
Estimated current stack usage: 7.62938 MB
User CPU Time: 0.6271 seconds
System CPU Time: 0.15969 seconds
Estimated current stack usage: 7.70567 MB
User CPU Time: 0.6271 seconds
System CPU Time: 0.16167 seconds
Estimated current stack usage: 7.78196 MB
User CPU Time: 0.6271 seconds
System CPU Time: 0.16349 seconds
Estimated current stack usage: 7.85826 MB
User CPU Time: 0.6271 seconds
System CPU Time: 0.16513 seconds
Estimated current stack usage: 7.93455 MB
Segmentation fault (core dumped)
dmaks@Ubuntu-22:~/HDA/BS/p1/bs_p1$
```

Bild 6, Stack Limit überschritten (*funcRec*)

Wert Stacks wird auch mithilfe von Funktion *availableStackProvider* als „STACK at disposal“ zurückgeliefert, diese Funktion benutzt weiterhin die *getrlimit* Funktion um Ergebnis zu liefern.

Unser Experiment, zeigt, dass wir 8MB Stack zur Verfügung haben.

b)

Dokumentieren Sie die von Ihrem Programm benötigte user bzw. system CPU time und deren Verhältnis in Abhängigkeit von n bzw. m (s.o.). Dazu können Sie eine Tabelle oder ein Diagramm verwenden.

funcMem

Größe Arrays ist: 1410065000

Output Frequency	User CPU Time (sec)	System CPU Time (sec)
100	104	349
1000	31	50
10000	24	15
100000	22	13
1000000	20	11
10000000	18	10
100000000	16	9

Tabelle 1, funcMem Ergebnisse (gerundet)

funcRec

Anzahl der Aufrufe ist: 104600

Output Frequency	User CPU Time (sec)	System CPU Time (sec)
1	0.346	1.647
10	0.465	0.169
100	0.498	0.331
1000	0.611	0.136

Tabelle 2, funcRec Ergebnisse