



## **Laporan Praktikum Manajemen Main Memory**

### **Praktikum Sistem Operasi**

#### **PTI Kelas C**

#### **Nama Mahasiswa:**

Ahmad Nauval Syahputra      215150601111021

#### **Asisten Praktikum:**

Iqbal Biondy      205150601111009

#### **Dosen:**

Faizatul Amalia, S.Pd.,M.Pd



**Program Studi Pendidikan Teknologi Informasi**

**Jurusan Sistem Informasi**

**Universitas Brawijaya**

**2022**

1. Langkah Praktikum

- a. Isikan Langkah-langkah dan screenshot sesuai dengan modul praktikum
- b. Setiap screenshot wajib disertai nama dan nim

2. Latihan Praktikum

- a. Isikan Latihan Praktikum dan screenshot sesuai dengan modul praktikum
- b. Setiap screenshot wajib disertai nama dan nim

3. Kesimpulan

Buatlah dalam bentuk paragraph dalam menjawab pertanyaan ini

- a. Jelaskan hubungan antara proses dan thread
- b. Jelaskan perbedaan dari singlethread dan multithread
- c. Sebutkan dan jelaskan macam-macam multithreading model

4. Referensi

Buatlah referensi dengan format sesuai dengan referensi pada penulisan skripsi <<http://file-filkom.ub.ac.id/fileupload/assets/upload/filemanager/Skripsi/PanduanSkripsiFilkom-v3.0.pdf>>

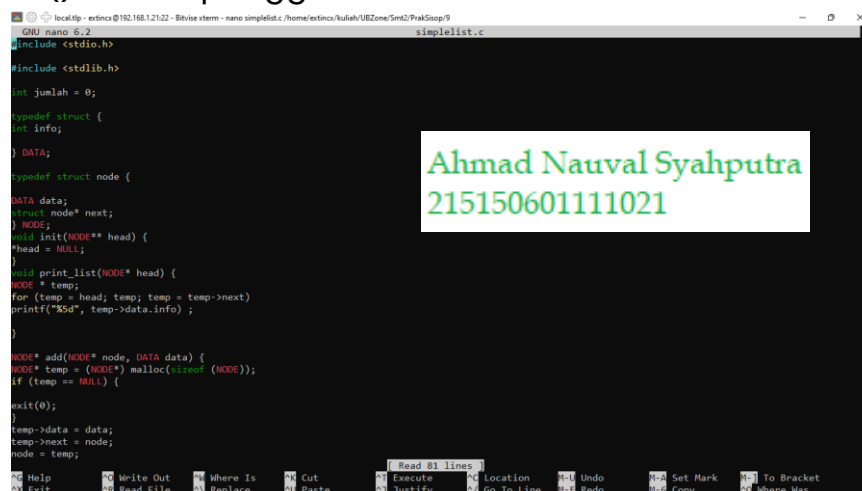
## 1. Langkah Praktikum

### a. Jelaskan fungsi dan return value dari malloc() dan free()

Fungsi malloc() adalah fundamental dari alokasi memori, memiliki sebutan lengkap Memory Allocation dengan sintaks dasar `void malloc(int byte_size);` Dengan banyaknya byte yang akan dipesan dinyatakan dalam bentuk parameter fungsi. Dengan kembalian (return value) berupa pointer tanpa tipe (pointer to void) yang menunjuk ke buffer (tempat penyimpanan sementara) Pointer perlu di ubah dahulu agar bisa mengakses data di buffer. Malloc berguna pada byte ruang memory yang belum diinisialisasi, dan apabila tidak terpenuhi akan mengembalikan nilai NULL.

Fungsi free() adalah fungsi dari pemrograman C untuk membebaskan memori yang telah dipakai dalam fungsi malloc() atau calloc() dengan sintaks dasar `void free(void memblock)`. Free tidak mengembalikan data ( no return value ). Digunakan untuk menghindari pemborosan memori atau memory leak sehingga perlu direalokasikan ruang memorinya sebelum digunakan lagi.

### b. Jelaskan jalannya simplelist.c, fungsi add() dan free\_list()? Bagaimana bila free\_list() tidak dipanggil?



```
GNU nano 2.2 simplelist.c
#include <stdio.h>
#include <stdlib.h>

int jumlah = 0;

typedef struct {
    int info;
} DATA;

typedef struct node {
    DATA data;
    struct node* next;
} NODE;

void init(NODE** head) {
    *head = NULL;
}

void print_list(NODE* head) {
    NODE* temp;
    for (temp = head; temp != temp->next; temp = temp->next)
        printf("%5d", temp->data.info);
}

NODE* add(NODE* node, DATA data) {
    NODE* temp = (NODE*) malloc(sizeof(NODE));
    if (temp == NULL) {
        exit(0);
    }
    temp->data = data;
    temp->next = node;
    node = temp;
}
```

Program membuat variable integer i, lalu objek dari NODE dan DATA. Kemudian menginisiasi nilai ke objek element serta memasukkan ke head, kemudian ditampilkan dengan fungsi print\_list(). Lalu akhir

program menghapus entry pada node dan mengeluarkan exit code sebagai signal exit.

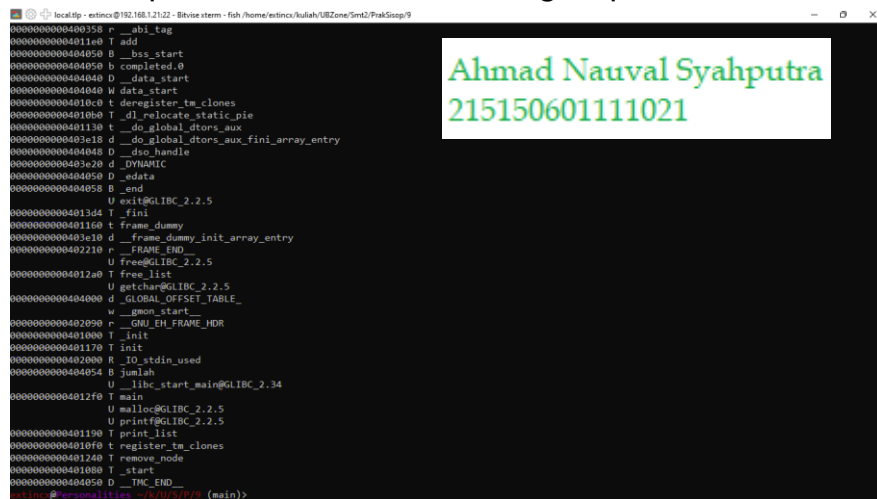
Add() berfungsi menambahkan (node dan data) ke array.

Free\_list() berfungsi menghapus entry NULL pada head.

Apabila tidak dipanggil maka entry malloc sebelumnya tetap ada, tetap berada pada memory dan Pointer tidak ke posisi semula.

## 2. Latihan Praktikum

a. Compile file simplelist.c dan analisa dengan perintah *nm*!



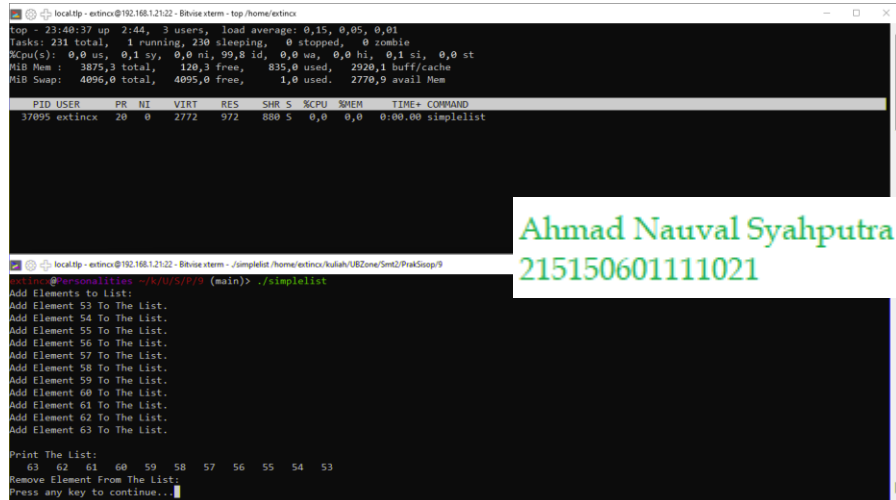
```
0000000000400350 r __abi_tag
00000000004011e0 t __add
0000000000404050 B __bss_start
0000000000404050 b __completed.0
0000000000404040 D __data_start
0000000000404040 W __data_start
00000000004010c0 t __deregister_tm_clones
00000000004010b0 t __dl_relocate_static_pie
0000000000401130 t __do_global_ctors_aux
0000000000403e18 d __do_global_ctors_aux_fini_array_entry
0000000000404048 D __dso_handle
0000000000402e70 d __dynamic
0000000000404050 D __edata
0000000000404058 B __end
0000000000401340 t __exit@GLIBC_2.2.5
0000000000401160 t __frame_dummy
0000000000403e10 d __frame_dummy_init_array_entry
0000000000402710 r __FRAME_END__
0000000000402710 U __free@GLIBC_2.2.5
00000000004012a0 t __free_list
0000000000404000 d __GLOBAL_OFFSET_TABLE__
0000000000402090 r __gmon_start__
0000000000401000 t __init
0000000000401170 t __init
0000000000402000 R __IO_stdin_used
0000000000404054 B __jmplsh
00000000004012f0 t __libc_start_main@GLIBC_2.3.4
0000000000401240 t __main
0000000000401190 U __malloc@GLIBC_2.2.5
0000000000401190 U __printf@GLIBC_2.2.5
00000000004010f0 t __register_tm_clones
0000000000401240 t __remove_node
0000000000401000 t __start
0000000000404050 D __TMC_END__
nm -C /home/estirco/Isiah/UBZone/Sm2/PrakSisop/9 (main)>
```

Perintah *nm* menunjukkan alamat symbolic dan relocatable pada simplelist yang tercompile. Nomor byte untuk alamat relocatable dan huruf untuk symbolic.

Tidak semua variable memiliki alamat symbolic, contohnya *int i* tidak ada pada *nm*.

Variable *jumlah* dan *heap* berbeda, *jumlah* dialokasikan sedangkan *heap* tidak pada *nm*.

b. Jalankan simplelist.c dan analisa pemakaiannya dengan perintah *top*!



The image contains two terminal screenshots. The top screenshot shows the output of the `top` command, displaying system statistics and a table of running processes. The bottom screenshot shows the execution of a program named `simplelist`, which adds elements to a list and prints the list.

```
top - 23:40:37 up 2:44, 3 users, load average: 0.15, 0.05, 0.01
Tasks: 231 total, 1 running, 230 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.0 us, 0.1 sy, 0.0 ni, 99.8 id, 0.0 wa, 0.0 hi, 0.1 si, 0.0 st
Mem Mem : 3875.3 total, 120.3 free, 835.0 used, 2920.1 buff/cache
Mem Swap: 4095.0 total, 4095.0 free, 1.0 used, 2770.9 avail Mem

  PID USER      PR  NI  VIRT  RES  SHR S %CPU  MEM%   TIME+  COMMAND
 37095 extincx  20   0   2772   972   880 S   0.0    0.0   0:00.00 simplelist

Ahmad Nauval Syahputra
215150601111021

extincx@Personalities: ~/K/ID/S/P/9 (main)> ./simplelist
Add Elements to List:
Add Element 53 To The List.
Add Element 54 To The List.
Add Element 55 To The List.
Add Element 56 To The List.
Add Element 57 To The List.
Add Element 58 To The List.
Add Element 59 To The List.
Add Element 60 To The List.
Add Element 61 To The List.
Add Element 62 To The List.
Add Element 63 To The List.
Print The List:
63 62 61 60 59 58 57 56 55 54 53
Remove Element From The List:
Press any key to continue...
```

Menunjukkan priority 20, nice 0, virtualisasi 2772, ressource 972 pada perintah `top`.

Penggunaan memory 0 persen pada saat program pada state Exception Press anykey...

Setelah menambah sampai 1000 list, hanya menambah penggunaan ressource namun memory tidak diketahui karena program sangat ringan.

c. Jalankan simplelist.c dan analisa pemakaiannya dengan perintah *pmap*!

```
-V, --version output version information and exit

For more details see pmap(1).
extinct@Personalities ~/K/U/S/P/9 (main) [1] ./pmap simplelist
Command './pmap' is available in the following places
 * /usr/bin:/pmap
 * /usr/bin:/pmap
./pmap: command not found
extinct@Personalities ~/K/U/S/P/9 (main) [127] http
extinct@Personalities ~/K/U/S/P/9 (main) [1] pmap -x 37095
37095: ./simplelist
Address      Kbytes    RSS    Dirty Mode Mapping
0000000000400000 4 4 0 r---- simplelist
0000000000401000 4 4 0 r---- simplelist
0000000000402000 4 4 0 r---- simplelist
0000000000403000 4 4 4 r---- simplelist
0000000000404000 4 4 4 r---- simplelist
0000000000405000 132 4 4 r---- [ anon ]
00007f9c232a000 12 8 0 r---- [ anon ]
00007f9c232c000 160 160 0 r---- libc.so.6
00007f9c232d000 1620 812 0 r---- libc.so.6
00007f9c232e000 352 128 0 r---- libc.so.6
00007f9c232f000 16 16 36 r---- libc.so.6
00007f9c2330000 8 8 8 r---- libc.so.6
00007f9c2331000 52 20 20 r---- [ anon ]
00007f9c2332000 8 4 4 r---- [ anon ]
00007f9c2333000 8 8 0 r---- ld-linux-x86-64.so.2
00007f9c2334000 160 160 0 r---- ld-linux-x86-64.so.2
00007f9c2335000 44 44 0 r---- ld-linux-x86-64.so.2
00007f9c2336000 8 8 8 r---- ld-linux-x86-64.so.2
00007f9c2337000 8 8 8 r---- ld-linux-x86-64.so.2
00007f9c2338000 132 16 36 r---- [ stack ]
00007f9c2339000 16 0 0 r---- [ anon ]
00007f9c233a000 8 4 0 r---- [ anon ]
00007f9c233b000 4 0 0 r---- [ anon ]
-----
total kB      2776 1436 100
extinct@Personalities ~/K/U/S/P/9 (main)>

0000000000402210 r _FRAME_END
00000000004012a0 U free@GLIBC_2.2.5
00000000004012a0 U free_list
0000000000404000 U getchar@GLIBC_2.2.5
0000000000404000 d _GLOBAL_OFFSET_TABLE_
0000000000402090 w _wcon_start
0000000000404000 r _GLOTH_FRAME_HDR
0000000000401000 T _init
0000000000401170 T init
0000000000402000 R _IO_stdin_used
0000000000404004 0 jumlah
0000000000401190 U _libc_start_main@GLIBC_2.34
00000000004012f0 T main
0000000000402250 U malloc@GLIBC_2.2.5
00000000004012a0 U printf@GLIBC_2.2.5
0000000000401190 T print_list
00000000004010f0 T register_tm_clones
00000000004012a0 U remove_node
0000000000402000 T _start
0000000000404050 D _TMC_END
extinct@Personalities ~/K/U/S/P/9 (main)> ./simplelist.c
Add Element 53 To The List.
Add Element 54 To The List.
Add Element 55 To The List.
Add Element 56 To The List.
Add Element 57 To The List.
Add Element 58 To The List.
Add Element 59 To The List.
Add Element 60 To The List.
Add Element 61 To The List.
Add Element 62 To The List.
Add Element 63 To The List.
Print The List:
63 62 61 60 59 58 57 56 55 54 53
Remove Element From The List:
Press any key to continue...
```

Ahmad Nauval Syahputra  
215150601111021

Proses selalu dialokasikan dengan baik dengan dialokasikan pada buffer, stack, linker, dan anon.

Kolom nya ada 4, pertama untuk Memory address, kedua untuk Besaran byte, ketiga untuk Ressource, ke empat untuk dirty access.

```
Print The List:
63 62 61 60 59 58 57 56 55 54 53
Remove Element From The List:
Press any key to continue...
extinct@Personalities ~/K/U/S/P/9 (main)> http
extinct@Personalities ~/K/U/S/P/9 (main) [1] http
extinct@Personalities ~/K/U/S/P/9 (main) [1] sudo pmap -x 39317
[sudo] password for extinct:
39317: ./simplelist
Address      Kbytes    RSS    Dirty Mode Mapping
0000558498bca000 4 4 0 r---- simplelist
0000558498bd0000 4 4 0 r---- simplelist
0000558498bd70000 4 4 0 r---- simplelist
0000558498bd71000 4 4 4 r---- simplelist
0000558498bd72000 4 4 4 r---- simplelist
00005584991a1000 132 32 32 r---- [ anon ]
00007f83130a000 12 8 8 r---- [ anon ]
00007f83130f1000 160 160 0 r---- libc.so.6
00007f831310000 1620 860 0 r---- libc.so.6
00007f831312a000 352 128 0 r---- libc.so.6
00007f8313130000 16 16 16 r---- libc.so.6
00007f8313130a000 8 8 8 r---- libc.so.6
00007f8313130c000 52 20 20 r---- [ anon ]
00007f8313132000 8 4 4 r---- [ anon ]
00007f8313132000 8 8 0 r---- ld-linux-x86-64.so.2
00007f8313132d000 168 168 0 r---- ld-linux-x86-64.so.2
00007f83131357000 44 44 0 r---- ld-linux-x86-64.so.2
00007f8313136000 8 8 8 r---- ld-linux-x86-64.so.2
00007f8313135000 8 8 8 r---- ld-linux-x86-64.so.2
00007f8313135000 132 16 16 r---- [ stack ]
00007f831312a000 16 0 0 r---- [ anon ]
00007f831312a000 8 4 0 r---- [ anon ]
00007f831312a000 4 0 0 r---- [ anon ]
-----
total kB      2776 1512 128
extinct@Personalities ~/K/U/S/P/9 (main)> http
extinct@Personalities ~/K/U/S/P/9 (main) [1] sudo pmap -x 39676
39676: ./simplelist
Address      Kbytes    RSS    Dirty Mode Mapping
00005583c286e000 4 4 0 r---- simplelist
00005583c286f000 4 4 0 r---- simplelist
00005583c2870000 4 4 0 r---- simplelist
00005583c2871000 4 4 4 r---- simplelist
00005583c2872000 4 4 4 r---- simplelist
00005583c2872000 31284 31252 31252 r---- [ anon ]
00007fcd17fb000 12 8 8 r---- [ anon ]
00007fcd17fb000 160 160 0 r---- libc.so.6
00007fcd17fd000 1520 800 0 r---- libc.so.6
00007fcd17fb000 352 128 0 r---- libc.so.6
00007fcd17f13000 16 16 16 r---- libc.so.6
00007fcd17f17000 8 8 8 r---- libc.so.6
00007fcd17f19000 52 20 20 r---- [ anon ]
00007fcd17f36000 8 4 4 r---- [ anon ]
00007fcd17f38000 8 8 0 r---- ld-linux-x86-64.so.2
00007fcd17f3a000 160 160 0 r---- ld-linux-x86-64.so.2
00007fcd17f3a000 44 44 0 r---- ld-linux-x86-64.so.2
00007fcd17f3a000 8 8 8 r---- ld-linux-x86-64.so.2
00007fcd17f3a000 8 8 8 r---- ld-linux-x86-64.so.2
00007fcd17f3a000 132 12 12 r---- [ stack ]
00007fcd17f3a000 16 0 0 r---- [ anon ]
00007fcd17f3a000 8 4 0 r---- [ anon ]
00007fcd17f3a000 4 0 0 r---- [ anon ]
-----
total kB      33928 32676 31344
extinct@Personalities ~/K/U/S/P/9 (main)>
```

```
00007f8313357000 44 44 0 r---- ld-linux-x86-64.so.2
00007f8313358000 8 8 8 r---- ld-linux-x86-64.so.2
00007f8313350000 8 8 8 r---- ld-linux-x86-64.so.2
00007f8313350000 132 16 16 r---- [ stack ]
00007f8313350000 16 0 0 r---- [ anon ]
00007f8313350000 8 4 0 r---- [ anon ]
00007f8313350000 4 0 0 r---- [ anon ]
-----
total kB      2776 1512 128
extinct@Personalities ~/K/U/S/P/9 (main)> http
extinct@Personalities ~/K/U/S/P/9 (main) [1] sudo pmap -x 39676
39676: ./simplelist
Address      Kbytes    RSS    Dirty Mode Mapping
00005583c286e000 4 4 0 r---- simplelist
00005583c286f000 4 4 0 r---- simplelist
00005583c2870000 4 4 0 r---- simplelist
00005583c2871000 4 4 4 r---- simplelist
00005583c2872000 4 4 4 r---- simplelist
00005583c2872000 31284 31252 31252 r---- [ anon ]
00007fcd17fb000 12 8 8 r---- [ anon ]
00007fcd17fb000 160 160 0 r---- libc.so.6
00007fcd17fd000 1520 800 0 r---- libc.so.6
00007fcd17fb000 352 128 0 r---- libc.so.6
00007fcd17f13000 16 16 16 r---- libc.so.6
00007fcd17f17000 8 8 8 r---- libc.so.6
00007fcd17f19000 52 20 20 r---- [ anon ]
00007fcd17f36000 8 4 4 r---- [ anon ]
00007fcd17f38000 8 8 0 r---- ld-linux-x86-64.so.2
00007fcd17f3a000 160 160 0 r---- ld-linux-x86-64.so.2
00007fcd17f3a000 44 44 0 r---- ld-linux-x86-64.so.2
00007fcd17f3a000 8 8 8 r---- ld-linux-x86-64.so.2
00007fcd17f3a000 8 8 8 r---- ld-linux-x86-64.so.2
00007fcd17f3a000 132 12 12 r---- [ stack ]
00007fcd17f3a000 16 0 0 r---- [ anon ]
00007fcd17f3a000 8 4 0 r---- [ anon ]
00007fcd17f3a000 4 0 0 r---- [ anon ]
-----
total kB      33928 32676 31344
extinct@Personalities ~/K/U/S/P/9 (main)>
```

Penggunaan resource lebih besar untuk penambahan node, sangat signifikan ketika perbedaan sangat besar, juga menambah waktu pemrosesan data.

Resource diperlukan pada data yang besar berbeda pada penggunaan dengan data kecil

Data kecil cenderung hemat daya dan data besar memerlukan runtime tinggi dan besar.

### 3. Kesimpulan

Penggunaan memory beraneka ragam namun mereka pada dasarnya memiliki 3 kunci utama yaitu address symbolic , reallocation, dan absolute.

Memegang peranan alokasi program di memory dengan protokoler melalui 3 fase yaitu Kompilasi, lalu Loading, dan kemudian Runtime.

Program memiliki alamat Physical dan Logical perbedaannya pada yang menanganinya yaitu memory dan cpu. Dan pada runtime maka ditangani oleh MMU pada Sistem Operasi.