



سؤال ۱. استفاده از فراخوانی‌های سیستمی malloc و free

۱. در صورتی که س‌ایز آن صفر نباشد، یک نشان‌گر^۱ برمی‌گرداند و در صورتی که س‌ایز برابر با صفر باشد، یا نشان‌گر خالی و یا یک نشان‌گر یکتا که به تابع free ارسال می‌شود، برگردانده می‌شود؛ در غیر این صورت (س‌ایز کم‌تر از صفر)، یک نشان‌گر خالی همراه با کد خطا نمایش داده می‌شود.

۲. اختصاص مقادیر:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

struct MyStruct {
    int a;
    int b;
    char name[20];
};

struct MyStruct *instance;

int main()
{
    instance = (struct MyStruct *) malloc(sizeof(struct MyStruct));
    instance -> a = 4;
    instance -> b = 5;
    strcpy(instance -> name, "Mostafa Ghadimi");
    printf("a: \t%d\n", instance -> a);
    printf("b: \t%d\n", instance -> b);
    printf("name: \t%s\n", instance -> name);
}
```

۳. برای آزاد کردن حافظه، خط زیر را به کد بالا اضافه می‌کنیم:

```
free(instance);
```

^۱pointer

سؤال ۲. مشاهده‌ی وضعیت حافظه‌ی پردازها

۱. وضعیت حافظه‌ی پردازها:

```
> ps -o user,vsz,rss,pmem,fname -e
USER      VSZ    RSS %MEM COMMAND
root      226304 10144 1.0 systemd
root        0      0 0.0 kthreadd
root        0      0 0.0 kworker/
root        0      0 0.0 loop0
root      179888 9052 1.0 thermalld
avahi      48272 4524 0.0 avahi-da
gdm        114452 2884 0.0 (sd-pam)
gdm        190688 5328 0.0 gdm-wayl
gdm         50352 3656 0.0 dbus-dae
mostafa    271036 5664 0.0 gsd-mous
mostafa    501660 9328 1.0 gsd-prin
```

نکته: خروجی بالا فقط برای نمونه آورده شده است و طول خروجی واقعی بسیار بیش‌تر از حالت فعلی است.

۲. توضیحات مربوط به اطلاعات هر کدام از ستون‌ها در ادامه آورده شده است.

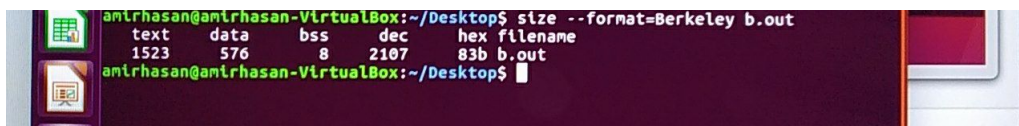
- **user**: نام کاربری را نشان می‌دهد.
- **vsz**: سایز حافظه مجازی اختصاص یافته به پردازه را به کیلوبایت نشان می‌دهد.
- **rss**: resident set size و اندازه‌ای از حافظه‌ی فیزیکی swap نشده را نمایش می‌دهد که تسک آن استفاده کرده است.
- **pmem**: نسبت اندازه‌ی resident set size به حافظه‌ی فیزیکی مورد استفاده‌ی پردازه را نشان می‌دهد.
- **fname**: ۸ بایت اول base name مربوط به فایل اجرایی پردازه را نشان می‌دهد.

سؤال ۳. اجزای حافظه‌ی یک پردازنده

۱. محل قرارگیری دستور ls بر روی دیسک

```
mostafa@mostafa-UX303UB:~$ which ls  
/bin/ls
```

۲. این دستور، مقدار حافظه‌ی اختصاص‌یافته به heap و stack را نمایش نمی‌دهد.



```
amirhasan@amirhasan-VirtualBox:~/Desktop$ size --format=Berkeley b.out  
text    data     bss      dec      hex filename  
1523     576         8    2107     83b b.out  
amirhasan@amirhasan-VirtualBox:~/Desktop$
```

سؤال ۴. اشتراک حافظه

۱. کتابخانه‌های مشترک دستور ls

```
> ldd /bin/ls
linux-vdso.so.1 (0x00007ffcce5bc000)
libselinux.so.1 => /lib/x86_64-linux-gnu/libselinux.so.1 (0x00007f33050d1000)
libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f3304ce0000)
libpcre.so.3 => /lib/x86_64-linux-gnu/libpcre.so.3 (0x00007f3304a6e000)
libdl.so.2 => /lib/x86_64-linux-gnu/libdl.so.2 (0x00007f330486a000)
/lib64/ld-linux-x862.so.64- (0x00007f330551b000)
libpthread.so.0 => /lib/x86_64-linux-gnu/libpthread.so.0 (0x00007f330464b000)
```

۲. کتابخانه‌های مشترک دستور nano

```
> ldd /bin/nano
linux-vdso.so.1 (0x00007ffe442a4000)
libncursesw.so.5 => /lib/x86_64-linux-gnu/libncursesw.so.5 (0x00007f70b30f2000)
libtinfo.so.5 => /lib/x86_64-linux-gnu/libtinfo.so.5 (0x00007f70b2ec8000)
libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f70b2ad7000)
libdl.so.2 => /lib/x86_64-linux-gnu/libdl.so.2 (0x00007f70b28d3000)
/lib64/ld-linux-x862.so.64- (0x00007f70b355d000)
```

کتابخانه‌های مشترک دستور pwd

```
> ldd /bin/pwd
linux-vdso.so.1 (0x00007ffe9e7e9000)
libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f8755132000)
/lib64/ld-linux-x862.so.64- (0x00007f875572c000)
```

سؤال ۵. آدرس بخش‌های مختلف حافظه‌ی پردازنده

۱. آدرس نماد^۲ها، پایان بخش‌های برنامه‌های مختلف را نمایش می‌دهد.
کد برنامه اجرا شده:

```
#include <stdio.h>
#include <stdlib.h>

int
main(int argc, char *argv[])
{
    printf("First address past:\n");
    printf("    program text (etext)      %10p\n", &etext);
    printf("    initialized data (edata)    %10p\n", &edata);
    printf("    uninitialized data (end)      %10p\n", &end);

    exit(EXIT_SUCCESS);
}
```

خروجی:

```
First address past:
    program text (etext)      0x5587a52d27ad
    initialized data (edata)  0x5587a54d3010
    uninitialized data (end)  0x5587a54d3018
```

۲. با توجه به خروجی قسمت ۱، با شکل مطابقت دارد.

^۲symbol