Võ Minh Long
MSSV: 1812951

Problem 2 (5 points) Write a short essay which summarizes the knowledge of process's data segment to answer for these question

Thông thường, một Unix Process sẽ được chia thành nhiều phân đoạn khác nhau (segments): code segment, data segment, BSS (block started by symbol), and stack segment, heap segment,...

Dưới góc độ của người sử dụng (user mode) thì mỗi segments có một đặc điểm và chức năng riêng:

- Code segment: chứa mã (code) thực thi của chương trình.
- Data segment: chứa các biến đã được khởi tạo (có giá trị cụ thể trước lúc thực thi).
- BSS: chứa các biến được khởi tạo những chưa có giá trị cụ thể. Giá trị này sẽ được sinh ra trong lúc thực thi chương trình.
- Stack segment: là nơi cấp phát vùng nhớ cho các biến cục bộ, hoặc lưu địa chỉ trả về của các function trước khi thực hiện lệnh gọi hàm.
- Heap segment: nơi chúng ta thực hiện cấp phát động (dynamic memory allocation)

Thông thường, chúng ta dùng lệnh **malloc(size_t size)** và **free(* ptr)** để xin cấp phát một vùng nhớ trên C khi có nhâu cầu và giải phóng vùng nhớ đó khi không có nhu cầu sử dụng nữa. Ngoài ra chúng ta còn có thể hiện thực, sử dụng hai hàm **aligned_malloc(size_t size, size_t align)** và **aligned_free(*ptr)** để cấp phát vùng nhớ theo một điều kiện chắc chẽ hơn (bội số của align, align phải là lũy thừa của 2) và giải phóng nó khi hết nhu cầu sử dụng.

1. ***Khi nào chúng ta nên sử dụng aligned_ malloc() thay vì malloc()?***
   - Chúng ta thường dùng khi muốn tối ưu hóa phù hợp với các bộ nhớ cache, page, hoặc theo yêu cầu phần cứng, yêu cầu của hệ điều hành,...
   - Aligned_malloc() được sử dụng rộng rãi trong:
     - Hệ thống nhúng.
     - Quá trình ánh xạ vùng nhớ vật lí sang vùng nhớ ảo và ngược lại (DMA).
     - Xử lí truy cập và tối ưu hóa bộ nhớ: địa chỉ không căn chỉnh (unaligned address) đối với nhiều lệnh đọc bộ nhớ (multiple read instruction) và địa chỉ cần căn chỉnh (aligned address) đối với lệnh đọc bộ nhớ đơn (single read instruction)
2. ***Làm sao để tăng kích thước của heap đối với môt process đang chạy?***
   - Hàm malloc thực chất là gọi hai system call **brk()** và **sbrk()** (có thể còn 1 số hàm khác).
   - Trong đó **sbrk(intptr_t increment)** dùng để thay đổi kích thước vùng nhớ đã cấp phát.
     - **intptr_t** là kiểu dữ liệu unsigned integer để lưu dữ liệu con trỏ (kích thước con trỏ)
     - nếu giá trị này dương thì vùng nhớ sẽ được mở rộng (dịch lên trên), ngược lại, giá trị này âm vùng nhớ sẽ bị thu hồi (dịch xuống dưới).
   - Dễ dàng thấy rằng, sbrk() là một sự lựa chọn nếu muốn tăng kích thước của heap đối với một process đang chạy. Trong thực tế, chúng ta chỉ cần gọi lại lệnh **malloc** và phần còn lại **heap managerment** sẽ lo giúp bạn.