Zadanie 1.

Brk i sbrk (pierwsze ustawia, drugie przesuwa o daną liczbę bajtów) przesuwa koniec *data segment*, więc można tylko allokować i zwalniać pamięć na końcu *data segment*. Mmap i munmap pozwala na allokację i deallokację w dowolnym miejscu.

sbrk: Załóżmy, że proces na początku zaallokował dużo pamięci. Potem części przestał używać, ale tę, którą zaallokował najpóźniej zostawił. Teraz nie można zwolnić tej nieużywanej pamięci.

mmap/munmap: Przydzielane fragmenty pamięci można allokować i zwalniać niezależnie, więc nie ma tego problemu.

Zadanie 2.

Fragmentacja wewnętrzna – Dopełnienia bloków, które nie są wykorzystane. Postają, bo mamy metadane dla struktur danych lub wyrównania adresów.

Fragmentacja zewnętrzna – Niewykorzystana przestrzeń pomiędzy zaalokowanymi obszarami pamięci jest zbyt mała, by można ją było wykorzystać.

Kompaktowanie – przesuwanie zajętych bloków pamięci tak, aby wyelminować fragmentację zewnętrzną. Allokator nie może tego robić, bo musiałby zmienić wartości wskaźników w programie użytkownika, do którego nie ma dostępu.

Fragmentacja zewnętrzna powstaje między innymi przez zwalnianie izolowanych bloków pamięci (otoczonych przez używane bloki. Ponadto powstaje też przez zmienne zachowanie programu. Allokator może działać lepiej jeśli allokacje i zwalnania pewnych obiektów tworzą powtarzalny wzór.

Zadanie 3.

Ramps – zużycie pamięci rośnie stopniowo. Program powoli buduje jakąś dużą strukturę danych (np. zapamiętuje wydarzenia).

Peeks – w kolejnych fazach działania proces buduje duże struktury danych, wykonuje na nich obliczenia, a następnie zwalnia tę pamięć zapamiętując tylko wyniki. *Plateaus* – proces w miarę szybko tworzy struktury danych, ale potem długo ich używa.

Bloki, które są podobnych rozmiarów najprawdopodobniej są tych samych typów. Bloki tych samych typów często są allokowane w podobnym czasie i są też zwalniane razem. Dlatego bloki tych samych rozmiarów powinno się umieszczać blisko siebie (będa miały podobny czas życia).

fist-fit – za każdym razem trzeba przeglądać mocno rozdrobnione bloki na początku.

next-fit – Słaba lokalność odwołań do pamięci. W praktycznych zastosowaniach fragmentacja jest gorsza niż w przypadku dwóch pozostałych. Są dwie odmiany: *address ordered* lub *LIFO*, w zależności od tego, w które miejsce listy dajemy zwolniony blok. W drugiej wersji zwolnienie bloku jest szybkie i nie wymaga dwustronnej kolejki, ale ta pierwsza pozwala na szybsze scalanie sąsiednich wolnych bloków i podowuje mniejszą fragmentację.

best-fit – dobrze ogranicza fragmentację, ale pesymistyczny czas działania jest duży.