**Pintos Project 0-2: Pintos Data Structure**

|  |  |
| --- | --- |
| 담당 교수 : | 김영재 |
| 학번 : | 20141196 |
| 이름 : | 김성희 |
|  |  |

**반드시 아래의 양식과 순서를 따라서 작성하기 바랍니다.**

1. **Additional Implementation**

|  |  |
| --- | --- |
| **Prototype** | void otherListFuncs(struct list \*\*l, char str[]) |
| **Parameter** | List, 명령어 문자열 |
| **Return** | None |
| **Function** | Dumpdata create list delete list 세가지 이외의 명령어에 대한 기능을 수행해 주는 함수 |
| **Prototype** | void otherBitmapFuncs(struct bitmap \*\*b, char str[]) |
| **Parameter** | 비트맵, 명령어 문자열 |
| **Return** | 없음 |
| **Function** | Dumpdata create bitmap delete bitmap 세가지 이외의 명령어에 대한 기능을 수행해 주는 함수 |
| **Prototype** | void otherHashFuncs(struct hash \*\*h, char str[]) |
| **Parameter** | Hash, 명령어 문자열 |
| **Return** | 없음 |
| **Function** | Dumpdata create delete 세가지 이외의 명령어에 대한 기능을 수행해 주는 함수 |

1. **List**

|  |  |
| --- | --- |
| **Prototype** | Void createList(struct list \*\*) |
| **Parameter** | 생성된 list가 담길 배열 주소(인덱스 주소) |
| **Return** | None |
| **Function** | List 생성 후 배열에 담아 줌 |
| **Prototype** | Void dumpdataList(struct list \*) |
| **Parameter** | 출력하고 싶은 list |
| **Return** | None |
| **Function** | List의 모든 데이터(원소) 출력 |
| **Prototype** | Void deleteList(struct list \*\*) |
| **Parameter** | 지우고 싶은 list를 담은 배열 주소(인덱스 주소) |
| **Return** | None |
| **Function** | List의 삭제 |
| **Prototype** | bool list\_less(const struct list\_elem\*, const struct list\_elem\*, void\*) |
| **Parameter** | 비교하고 싶은 list 원소의 list\_elem주소 두개와 보조 데이터 |
| **Return** | True or False |
| **Function** | 값 비교 후 앞의 list\_elem data 가 작으면 true 아니면 false |
| **Prototype** | Void list\_shuffle(struct list\*) |
| **Parameter** | Shuffle할 list |
| **Return** | None |
| **Function** | List 무작위 shuffle |
| **Prototype** | Static bool is\_sorted(struct list\_elem\*, struct list\_elem\*, list\_less\_func\*, void\*) |
| **Parameter** | List\_elem addr 2개, 대소 비교 함수 1개, 보조 데이터 1개 |
| **Return** | True or False |
| **Function** | 2개의 List\_elem 사이에 정렬이 되어 있는지 확인 |
| **Prototype** | static inline bool is\_head(struct list\_elem \*) |
| **Parameter** | List\_elem addr |
| **Return** | True false |
| **Function** | List\_elem이 head인지 판별 |
| **Prototype** | Static inline bool is\_interior(struct list\_elem \*) |
| **Parameter** | List\_elem addr |
| **Return** | True false |
| **Function** | List\_elem이 head, tail 이외의 노드인지 판별 (interior인지 판별) |
| **Prototype** | Static inline bool is\_tail(struct list\_elem \*) |
| **Parameter** | List\_elem addr |
| **Return** | True false |
| **Function** | List\_elem이 tail인지 판별 |
| **Prototype** | Void list\_init(struct list\*) |
| **Parameter** | List addr |
| **Return** | none |
| **Function** | List 초기화 |
| **Prototype** | Struct list\_elem\* list\_begin(struct list\*) |
| **Parameter** | List addr |
| **Return** | List\_elem addr |
| **Function** | List의 첫 interior주소 반환 |
| **Prototype** | Struct list\_elem\* list\_next(struct list\_elem\*) |
| **Parameter** | List\_elem addr |
| **Return** | List\_elem addr |
| **Function** | List\_elem의 다음 list\_elem 주소 반환 |
| **Prototype** | Struct list\_elem\* list\_end(struct list\*) |
| **Parameter** | List addr |
| **Return** | List\_elem addr |
| **Function** | List의 tail 주소 반환 |
| **Prototype** | Struct list\_elem\* list\_rbegin(struct list\*) |
| **Parameter** | List addr |
| **Return** | List\_elem addr |
| **Function** | List의 마지막 interior 주소 반환 |
| **Prototype** | Struct list\_elem\* list\_prev(struct list\_elem\*) |
| **Parameter** | List\_elem addr |
| **Return** | List\_elem addr |
| **Function** | List\_elem의 이전 list\_elem 주소 반환 |
| **Prototype** | Struct list\_elem\* list\_head(struct list\*) |
| **Parameter** | List addr |
| **Return** | List\_elem addr |
| **Function** | List의 head 주소 반환 |
| **Prototype** | Struct list\_elem\* list\_tail(struct list\*) |
| **Parameter** | List addr |
| **Return** | List\_elem addr |
| **Function** | List의 tail 주소 반환 |
| **Prototype** | Void list\_insert(struct list\_elem\*, struct list\_elem\*) |
| **Parameter** | 삽입할 위치의 뒤에 있는 list\_elem, 삽입할 list\_elem |
| **Return** | none |
| **Function** | 첫번째 List\_elem 인자 앞에 두번째 list\_elem 인자를 삽입 |
| **Prototype** | void  list\_splice (struct list\_elem \*before,               struct list\_elem \*first, struct list\_elem \*last) |
| **Parameter** | List element 3개 |
| **Return** | None |
| **Function** | Before 앞에 first 부터 last까지 삽입 |
| **Prototype** | void  list\_push\_front (struct list \*list, struct list\_elem \*elem) |
| **Parameter** | List와 list element |
| **Return** | None |
| **Function** | List의 앞에 element 삽입 |
| **Prototype** | void  list\_push\_back (struct list \*list, struct list\_elem \*elem) |
| **Parameter** | List와 list element |
| **Return** | None |
| **Function** | List 뒤에 list element 삽입 |
| **Prototype** | struct list\_elem \*  list\_remove (struct list\_elem \*elem) |
| **Parameter** | List element |
| **Return** | 삭제한 element의 다음 element |
| **Function** | Element 삭제 후 next element 반환 |
| **Prototype** | struct list\_elem \*  list\_pop\_front (struct list \*list) |
| **Parameter** | List |
| **Return** | 삭제한 List element |
| **Function** | List의 첫 interior 삭제 후 삭제한 element 반환 |
| **Prototype** | struct list\_elem \*  list\_pop\_back (struct list \*list) |
| **Parameter** | List |
| **Return** | 삭제한 list element |
| **Function** | List의 마지막 interior 삭제 후 반환ㄴ |
| **Prototype** | struct list\_elem \*  list\_front (struct list \*list) |
| **Parameter** | List |
| **Return** | List의 첫 interior List element |
| **Function** | List의 첫 interior List element 반환 |
| **Prototype** | struct list\_elem \*  list\_back (struct list \*list) |
| **Parameter** | List |
| **Return** | List의 마지막 interior list element |
| **Function** | List의 마지막 interior list element 반환 |
| **Prototype** | size\_t  list\_size (struct list \*list) |
| **Parameter** | List |
| **Return** | List 크기 |
| **Function** | List의 크기 반환 |
| **Prototype** | bool  list\_empty (struct list \*list) |
| **Parameter** | List |
| **Return** | True false |
| **Function** | List 비었는지 판별 (비었으면 true) |
| **Prototype** | static void  swap (struct list\_elem \*\*a, struct list\_elem \*\*b) |
| **Parameter** | List\_elem 주소 2개 |
| **Return** | None |
| **Function** | List\_elem 두개 바꾸기 |
| **Prototype** | void  list\_reverse (struct list \*list) |
| **Parameter** | List |
| **Return** | None |
| **Function** | List 거꾸로 뒤집기 |
| **Prototype** | static struct list\_elem \*  find\_end\_of\_run (struct list\_elem \*a, struct list\_elem \*b,                   list\_less\_func \*less, void \*aux) |
| **Parameter** | list\_elem 2개, 비교함수 1개, 보조 데이터 1개 |
| **Return** | List element |
| **Function** | list에서 a부터 b까지 중 비내림차순에 처음으로 위배되는 element 반환 |
| **Prototype** | static void  inplace\_merge (struct list\_elem \*a0, struct list\_elem \*a1b0,                 struct list\_elem \*b1,                 list\_less\_func \*less, void \*aux) |
| **Parameter** | List\_element 3개, 비교함수 1개, 보조 데이터 1개 |
| **Return** | None |
| **Function** | A0~a1b0 비내림차순 정렬, a1b0~b1 비내림차순 정렬 되어 있다고 가정하고 a0~b1을 정렬 (a0~a1b0, a1b0~b1 정렬되도록 합병) |
| **Prototype** | void  list\_sort (struct list \*list, list\_less\_func \*less, void \*aux) |
| **Parameter** | List, 비교함수, 보조 데이터 |
| **Return** | None |
| **Function** | List sorting |
| **Prototype** | void  list\_insert\_ordered (struct list \*list, struct list\_elem \*elem,                       list\_less\_func \*less, void \*aux) |
| **Parameter** | List, list element, 비교 함수, 보조 데이터 |
| **Return** | None |
| **Function** | Element를 list에 비내림차순이 되도록 삽입 |
| **Prototype** | void  list\_unique (struct list \*list, struct list \*duplicates,               list\_less\_func \*less, void \*aux) |
| **Parameter** | 원소 하나씩 가지게 만들 list 1개와 제거한 원소를 append할 duplicate list 1개. 비교 함수, 보조 데이터 각각 1개 |
| **Return** | None |
| **Function** | List의 중복 데이터 삭제, 삭제한 중복 데이터들 duplicates에 복사 |
| **Prototype** | struct list\_elem \*  list\_max (struct list \*list, list\_less\_func \*less, void \*aux) |
| **Parameter** | List, 비교함수, 보조데이터 |
| **Return** | List element |
| **Function** | List중 max data가진 element 반환 |
| **Prototype** | struct list\_elem \*  list\_min (struct list \*list, list\_less\_func \*less, void \*aux) |
| **Parameter** | List, 비교함수, 보조데이터 |
| **Return** | List element |
| **Function** | List 중 min data를 가진 element 반환 |

**III. Hash Table**

|  |  |
| --- | --- |
| **Prototype** | bool  hash\_init (struct hash \*h, hash\_hash\_func \*hash,  hash\_less\_func \*less, void \*aux) |
| **Parameter** | Hash, hash function, 비교함수, 보조데이터 |
| **Return** | True false |
| **Function** | Element 수와 bucket수를 0, 4로 초기화하고, hash function과 비교함수를 저장한다. Bucket에 메모리를 할당한다. (잘 되면 true NULL이면 false) |
| **Prototype** | void  hash\_clear (struct hash \*h, hash\_action\_func \*destructor) |
| **Parameter** | Hash, destructor function |
| **Return** | None |
| **Function** | Hash의 모든 element 삭제 |
| **Prototype** | void  hash\_destroy (struct hash \*h, hash\_action\_func \*destructor) |
| **Parameter** | Hash, destructor function |
| **Return** | 없음 |
| **Function** | Hash의 모든 element를 삭제하고 bucket도 비운다.(할당해제) |
| **Prototype** | struct hash\_elem \*  hash\_insert (struct hash \*h, struct hash\_elem \*new) |
| **Parameter** | Hash와 hash element |
| **Return** | Hash element |
| **Function** | Hash에 new element를 저장한다. 이 때 new element의 value와 동일한 value를 가진 element가 있다면 저장하지 않고 기존의 hash element를 반환한다. (없다면 NULL 반환) |
| **Prototype** | struct hash\_elem \*  hash\_replace (struct hash \*h, struct hash\_elem \*new) |
| **Parameter** | Hash와 hash element |
| **Return** | Hash element |
| **Function** | Hash에 new element를 저장한다. new element의 value와 동일한 value를 가진 element가 있는지 없는지는 상관하지 않는다. 단, 있다면 해당 hash element를, 없다면 NULL을 반환한다. |
| **Prototype** | struct hash\_elem \*  hash\_find (struct hash \*h, struct hash\_elem \*e) |
| **Parameter** | Hash와 hash element |
| **Return** | Hash element |
| **Function** | e hash element와 동일한 value를 가진 hash element를 반환한다. |
| **Prototype** | struct hash\_elem \*  hash\_delete (struct hash \*h, struct hash\_elem \*e) |
| **Parameter** | Hash와 hash element |
| **Return** | Hash element |
| **Function** | E hash element와 동일한 value를 가진 hash element를 hash에서 지우고 반환한다. |
| **Prototype** | void  hash\_apply (struct hash \*h, hash\_action\_func \*action) |
| **Parameter** | Hash와 action function |
| **Return** | None |
| **Function** | Hash의 모든 element에 action fucntion을 적용한다. |
| **Prototype** | void  hash\_first (struct hash\_iterator \*i, struct hash \*h) |
| **Parameter** | Hash interator, hash |
| **Return** | None |
| **Function** | Hash의 첫 bucket의 head를 hash iterator에 저장 |
| **Prototype** | struct hash\_elem \*  hash\_next (struct hash\_iterator \*i) |
| **Parameter** | Hash iterator |
| **Return** | Hash element |
| **Function** | Iterator로부터 다음 hash element 반환 |
| **Prototype** | struct hash\_elem \*  hash\_cur (struct hash\_iterator \*i) |
| **Parameter** | Hash iterator |
| **Return** | Hash element |
| **Function** | iterator로부터 현재 hash element 반환 |
| **Prototype** | bool  hash\_empty (struct hash \*h) |
| **Parameter** | Hash |
| **Return** | True false |
| **Function** | Hash에 hash element가 있다면 true 없다면 falses |
| **Prototype** | unsigned  hash\_bytes (const void \*buf\_, size\_t size) |
| **Parameter** | key, size |
| **Return** | HashValue |
| **Function** | key로부터 hashValue를 구해준다. |
| **Prototype** | unsigned  hash\_string (const char \*s\_) |
| **Parameter** | String key |
| **Return** | hashValue |
| **Function** | String key로부터 hashValue를 구해준다. |
| **Prototype** | unsigned  hash\_int (int i) |
| **Parameter** | Integer key |
| **Return** | hashValue |
| **Function** | Integer key로부터 hashValue를 구해준다. |
| **Prototype** | static struct list \*  find\_bucket (struct hash \*h, struct hash\_elem \*e) |
| **Parameter** | Hash element |
| **Return** | Bucket |
| **Function** | hash에서 Hash element가 어느 bucket에 담길지 알려준다. |
| **Prototype** | static struct hash\_elem \*  find\_elem (struct hash \*h, struct list \*bucket,  struct hash\_elem \*e) |
| **Parameter** | Hash, bucket, hash element |
| **Return** | Hash element |
| **Function** | Hash와 bucket 정보를 통해 hash element와 동일한 hash element를 찾아준다. |
| **Prototype** | static inline size\_t  turn\_off\_least\_1bit (size\_t x) |
| **Parameter** | Number |
| **Return** | Number |
| **Function** | Number x의 lowest-order인 bit 1 제거 (1->0) |
| **Prototype** | static inline size\_t  is\_power\_of\_2 (size\_t x) |
| **Parameter** | Number |
| **Return** | Number (1or0) |
| **Function** | Number x가 2의 제곱수인지 판별 |
| **Prototype** | static void  rehash (struct hash \*h) |
| **Parameter** | Hash |
| **Return** | 없음 |
| **Function** | Hashtable을 재설정 |
| **Prototype** | static void  insert\_elem (struct hash \*h, struct list \*bucket,   struct hash\_elem \*e) |
| **Parameter** | Hash, bucket, hash element |
| **Return** | None |
| **Function** | Hash의 bucket 맨 앞에 hash element 삽입 |
| **Prototype** | static void  remove\_elem (struct hash \*h, struct hash\_elem \*e) |
| **Parameter** | Hash, hash element |
| **Return** | None |
| **Function** | hash에서 hash element 제거 |
| **Prototype** | bool hash\_less(const struct hash\_elem \*a,  const struct hash\_elem \*b, void \*aux) |
| **Parameter** | Hash element X 2, 보조데이터 |
| **Return** | True false |
| **Function** | Hash element a 의 value가 더 작으면 true 아니면 false |
| **Prototype** | void destructor(struct hash\_elem \*a, void \*aux) |
| **Parameter** | Hash element, 보조데이터 |
| **Return** | 없음 |
| **Function** | free Hash element |
| **Prototype** | unsigned int hash\_func(const struct hash\_elem \*a, void \*aux) |
| **Parameter** | Hash element, 보조데이터 |
| **Return** | hashValue |
| **Function** | Hash element를 인자로 넣어서 hashValue를 얻는다. |
| **Prototype** | unsigned  hash\_int\_2 (int i) |
| **Parameter** | Number |
| **Return** | Unsigned Number |
| **Function** | Hash 함수 (int type key에 대한) |

**IV. Bitmap**

|  |  |
| --- | --- |
| **Prototype** | void dumpdataBitmap(struct bitmap \*b) |
| **Parameter** | 비트맵 |
| **Return** | 없음 |
| **Function** | 비트맵 출력 (2진수로) |
| **Prototype** | void deleteBitmap(struct bitmap \*\*b) |
| **Parameter** | 비트맵 |
| **Return** | 없음 |
| **Function** | 비트맵 삭제 |
| **Prototype** | void bitmap\_expand(struct bitmap \*b, size\_t size) |
| **Parameter** | 비트맵, 추가할 크기 |
| **Return** | 없음 |
| **Function** | Size 만큼 비트맵 공간 키우기 (비트맵 뒤에 size만큼 0붙이기) |
| **Prototype** | static inline size\_t  elem\_idx (size\_t bit\_idx) |
| **Parameter** | 비트맵 비트 위치 (상대적 위치) |
| **Return** | 비트맵 비트 위치 (절대적 위치) |
| **Function** | 원하는 비트의 실제 위치(인덱스) 반환 |
| **Prototype** | static inline elem\_type  bit\_mask (size\_t bit\_idx) |
| **Parameter** | 비트맵 위치(인덱스) |
| **Return** | 비트맵 |
| **Function** | 비트맵의 bit\_idx 위치 1로 만들기 |
| **Prototype** | static inline size\_t  elem\_cnt (size\_t bit\_cnt) |
| **Parameter** | 크기 |
| **Return** | 크기 |
| **Function** | 원하는 비트 수의 비트맵이 차지하는 실제 공간 크기 반환 |
| **Prototype** | static inline elem\_type  last\_mask (const struct bitmap \*b) |
| **Parameter** | 비트맵 |
| **Return** | 비트맵 |
| **Function** | 현재 사용 중인 비트들 모두 1 나머지 모두 0 (비트맵크기를 16으로 만들었으면 16개의 비트만 1, 나머진 0) |
| **Prototype** | struct bitmap \*  bitmap\_create (size\_t bit\_cnt) |
| **Parameter** | Size\_t 비트맵 크기 |
| **Return** | 비트맵 |
| **Function** | 비트맵 크기에 맞는 비트맵 생성 |
| **Prototype** | struct bitmap \*  bitmap\_create\_in\_buf (size\_t bit\_cnt, void \*block,                        size\_t block\_size ) |
| **Parameter** | Size\_t, void\*, size\_t |
| **Return** | 비트맵 |
| **Function** | Block\_size와 비트맵을 안전하게 생성 |
| **Prototype** | size\_t  bitmap\_buf\_size (size\_t bit\_cnt) |
| **Parameter** | 비트맵 크기 |
| **Return** | 비트맵 크기 |
| **Function** | 비트맵의 한계 크기 반환 |
| **Prototype** | void  bitmap\_destroy (struct bitmap \*b) |
| **Parameter** | 비트맵 |
| **Return** | 없음 |
| **Function** | 비트맵 제거(할당해제) |
| **Prototype** | size\_t  bitmap\_size (const struct bitmap \*b) |
| **Parameter** | 비트맵 |
| **Return** | 크기 |
| **Function** | 비트맵 크기 반환 |
| **Prototype** | void  bitmap\_set (struct bitmap \*b, size\_t idx, bool value) |
| **Parameter** | 비트맵, 위치, true false |
| **Return** | 없음 |
| **Function** | Idx 위치에 1 or 0 세팅 (true->1, false->0) |
| **Prototype** | void  bitmap\_mark (struct bitmap \*b, size\_t bit\_idx) |
| **Parameter** | 비트맵, 위치 |
| **Return** | 없음 |
| **Function** | Bit\_idx 위치에 1 세팅 |
| **Prototype** | Void  bitmap\_reset (struct bitmap \*b, size\_t bit\_idx) |
| **Parameter** | 비트맵, 위치 |
| **Return** | 없음 |
| **Function** | Bit\_idx 위치에 0 세팅 |
| **Prototype** | void  bitmap\_flip (struct bitmap \*b, size\_t bit\_idx) |
| **Parameter** | 비트앱, 위치 |
| **Return** | 없음 |
| **Function** | Bit\_idx 위치의 비트 반전 |
| **Prototype** | bool  bitmap\_test (const struct bitmap \*b, size\_t idx) |
| **Parameter** | 비트앱, 위치 |
| **Return** | True false |
| **Function** | 해당 위치가 무슨 비트인지 반환(1->true, 0->false) |
| **Prototype** | void  bitmap\_set\_all (struct bitmap \*b, bool value) |
| **Parameter** | 비트맵, true false |
| **Return** | 없음 |
| **Function** | 비트맵의 모든 비트 value로 세팅 (true->1, false->0) |
| **Prototype** | void  bitmap\_set\_multiple (struct bitmap \*b, size\_t start,    size\_t cnt, bool value) |
| **Parameter** | 비트맵, 시작위치, 개수, value |
| **Return** | 없음 |
| **Function** | 비트맵을 start위치부터 cnt개의 비트를 value로 바꿈 |
| **Prototype** | size\_t  bitmap\_count (const struct bitmap \*b, size\_t start,  size\_t cnt, bool value) |
| **Parameter** | 비트맵, 시작위치, 개수, value |
| **Return** | 개수 |
| **Function** | 비트맵의 start위치부터 cnt개의 비트 중 value와 같은 비트의 개수 반환 |
| **Prototype** | bool  bitmap\_contains (const struct bitmap \*b,  size\_t start, size\_t cnt, bool value) |
| **Parameter** | 비트맵, 시작위치, 개수, value |
| **Return** | True false |
| **Function** | 비트맵의 start위치부터 cnt개의 비트가 전부 value면 true 하나라도 아니면 false |
| **Prototype** | bool  bitmap\_any (const struct bitmap \*b, size\_t start, size\_t cnt) |
| **Parameter** | 비트맵, 시작위치, 개수 |
| **Return** | True false |
| **Function** | 비트맵의 start위치부터 cnt개의 비트 중 한 개라도 1이 있으면 true 아니면 false |
| **Prototype** | bool  bitmap\_none (const struct bitmap \*b, size\_t start, size\_t cnt) |
| **Parameter** | 비트맵, 시작위치, 개수 |
| **Return** | True false |
| **Function** | 비트맵의 start위치부터 cnt개의 비트가 모두 0이라면 true 1개라도 1이라면 false |
| **Prototype** | bool  bitmap\_all (const struct bitmap \*b, size\_t start, size\_t cnt) |
| **Parameter** | 비트맵, 시작위치, 개수 |
| **Return** | True false |
| **Function** | 비트맵의 start위치부터 cnt개의 비트가 모두 1이라면 true 1개라도 0이라면 false |
| **Prototype** | size\_t  bitmap\_scan (const struct bitmap \*b, size\_t start,  size\_t cnt, bool value) |
| **Parameter** | 비트맵, 시작위치, 개수, value |
| **Return** | 위치 |
| **Function** | 비트맵의 start위치부터 cnt개의 비트가 연속으로 value인 그룹들 중 가장 앞에 있는 그룹의 첫 비트 위치 반환 |
| **Prototype** | size\_t  bitmap\_scan\_and\_flip (struct bitmap \*b, size\_t start,   size\_t cnt, bool value) |
| **Parameter** | 비트맵, 시작위치, 개수, value |
| **Return** | 개수 |
| **Function** | 비트맵의 start위치부터 cnt개의 비트가 연속으로 value인 그룹들 중 가장 앞에 있는 그룹의 첫 비트 위치 반환. 그리고 가장 앞에 있는 그룹의 비트는 반전시킨다. |
| **Prototype** | size\_t  bitmap\_file\_size (const struct bitmap \*b) |
| **Parameter** | 비트맵 |
| **Return** | 크기 |
| **Function** | 비트맵 크기 |
| **Prototype** | bool  bitmap\_read (struct bitmap \*b, struct file \*file) |
| **Parameter** | 비트맵, 파일 |
| **Return** | True false |
| **Function** | 파일 성공적으로 읽으면 true 아니면 false |
| **Prototype** | bool  bitmap\_write (const struct bitmap \*b, struct file \*file) |
| **Parameter** | 비트맵, 파일 |
| **Return** | True, false |
| **Function** | 파일 성공적으로 쓰면 true 아니면 false |
| **Prototype** | void  bitmap\_dump (const struct bitmap \*b) |
| **Parameter** | 비트맵 |
| **Return** | 없음 |
| **Function** | 비트맵을 16진수로 출력 |