Pintos Project 0-2: Pintos Data Structure

담당 교수 : 박성용

학번 : 20161643

이름 : 정진원

반드시 아래의 양식과 순서를 따라서 작성하기 바랍니다.

I. Additional Implementation

Prototype	int get_num(char* idx)
Parameter	char* idx
Return	0~9 사이의 정수 값
Function	Parameter로 들어온 문자열의 마지막에 위치한 숫자를 정수 값으로 변환해서 반환해주는 함수이다. List, hash, bitmap의 경우 모두 10개까지 존재할 수 있고, 각각 몇 번째 원소인지 구해야 하는 경우가 자주 나와서 함수로 구현하였다.

Prototype	bool list_less_function(const struct list_elem* a, const struct list_elem* b, void *aux)
Parameter	const struct list_elem* a, const struct list_elem* b, void *aux
Return	True or false
Function	List entry function을 이용해서 parameter로 들어온 두 list element의 data의 크기를 비교하여 결과에 따라 참 또는 거짓을 반환한다.

Prototype	unsigned hash_hash_function(const struct hash_elem* a, void *aux)
Parameter	const struct hash_elem* a, void *aux
Return	hash_int(temp->data)
Function	Hash_item 함수와 Hash_int 함수를 이용해 hash element가 가지고 있는 데이터의 hash 값을 반환해준다.

Prototype	bool hash_less_function(const struct hash_elem* a, const struct hash_elem* b, void* aux)
Parameter	const struct hash_elem* a, const struct hash_elem* b, void* aux
Return	True or false
Function	hash entry function을 이용해서 parameter로 들어온 두 hash element의 data의 크기를 비교하여 결과에 따라 참 또는 거짓을 반환한다.

Prototype	void hash_action_destroy(struct hash_elem *a, void *aux)
Parameter	struct hash_elem *a, void *aux
Return	
Function	인자로 받은 hash element를 통해 hash entry function을 이용하여 hash item을 free해서 삭제시켜주는 함수이다.

Prototype	void hash_action_square(struct hash_elem* a, void *aux)
Parameter	struct hash_elem *a, void *aux
Return	
Function	인자로 받은 hash element를 포함하는 hash item을 hash entry 함수를 통해 얻은 뒤 data 값에 제곱을 해준다.

Prototype	void hash_action_triple(struct hash_elem* a, void *aux)
Parameter	struct hash_elem *a, void *aux
Return	
Function	인자로 받은 hash element를 포함하는 hash item을 hash entry 함수를 통해 얻은 뒤 data 값에 세제곱을 해준다.

II. List

Prototype	void list_init (struct list *list)
Parameter	struct list *list
Return	
Function	존재하는 list를 initialize 하여 head와 tail의 값을 설정해준다.

Prototype	static inline bool is_head (struct list_elem *elem)
Parameter	struct list_elem *elem
Return	True or false
Function	인자로 전달받은 element가 head에 해당하면 true를 반환하고 아니면 false를 반환한다.

Prototype	static inline bool is_interior (struct list_elem *elem)
Parameter	struct list_elem *elem
Return	True or false
Function	인자로 전달받은 element가 list 내부의 원소이면 true를 반환하고 아니면 false를 반환한다.

Prototype	static inline bool is_tail (struct list_elem *elem)
Parameter	struct list_elem *elem
Return	True or false
Function	인자로 전달받은 element가 list의 tail에 해당하면 true를 반환하고 아니면 false를 반환한다.

Prototype	struct list_elem * list_begin (struct list *list)
Parameter	struct list *list
Return	list->head.next
Function	List가 시작되는, 즉 head의 다음 element를 반환해준다.

Prototype	struct list_elem * list_next (struct list_elem *elem)
Parameter	struct list_elem *elem
Return	elem->next
Function	전달받은 list element의 다음 원소를 반환해준다.

Prototype	struct list_elem * list_end (struct list *list)
Parameter	struct list *list
Return	&list->tail
Function	전달받은 list의 마지막 원소를 return해주는 함수. 주로 list의 처음부터 끝까지 iterating 할 때 끝나는 지점을 확인하기 위해 사용된다.

Prototype	struct list_elem * list_rbegin (struct list *list)

Parameter	struct list *list
Return	list->tail.prev
Function	전달받은 list를 역순으로 했을 때, 시작 원소를 return 해주는 함수이다. 즉 리스트의 마지막 원소를 return해준다.

Prototype	struct list_elem * list_rend (struct list *list)
Parameter	struct list *list
Return	list_elem *
Function	List를 역순으로 했을 때 끝을 반환해주는 함수. 즉 list의 head를 반환해주는 함수. 주로 list를 거꾸로 iterate 할 때 사용된다.

Prototype	struct list_elem *list_prev (struct list_elem *elem)
Parameter	struct list *list
Return	elem->prev;
Function	전달받은 element의 전 원소를 return 해주는 함수.

Prototype	struct list_elem *list_head (struct list *list)
Parameter	struct list *list
Return	&list->head
Function	List가 null이 아닌 경우, list의 head를 반환해준다.

Prototype	struct list_elem * list_tail (struct list *list)
Parameter	struct list *list
Return	&list->tail;
Function	List가 null이 아닌 경우, list의 tail을 반환해준다.

Prototype	Void list_insert (struct list_elem *before, struct list_elem *elem)
Parameter	struct list_elem *before, struct list_elem *elem
Return	
Function	인자로 전달받은 before 원소의 전에, 전달받은 elem을 삽입하는 함수이다.

Prototype	Void list_splice (struct list_elem *before, struct list_elem *first, struct list_elem *last)
Parameter	struct list_elem *before, struct list_elem *first, struct list_elem *last
Return	
Function	전달 받은 first 원소부터 last 원소까지를 list에서 제거 한 뒤, 전달받은 before원소 전에 삽입하는 함수이다.

Prototype	Void list_push_front (struct list *list, struct list_elem *elem)
Parameter	struct list *list, struct list_elem *elem
Return	
Function	List의 처음에 전달받은 element를 삽입한다.

Prototype	Void list_push_back (struct list *list, struct list_elem *elem)
Parameter	struct list *list, struct list_elem *elem
Return	
Function	List의 마지에 전달받은 element를 삽입한다.

Prototype	struct list_elem * list_remove (struct list_elem *elem)
Parameter	struct list_elem *elem
Return	elem->next;
Function	전달 받은 element를 list에서 삭제한 뒤, 삭제한 다음의 원소를 반환한다.

Prototype	struct list_elem * list_pop_front (struct list *list)
Parameter	struct list *list
Return	front
Function	리스트의 첫 원소를 list에서 제거한 뒤, 그 원소를 반환해준다.

Prototype	struct list_elem *list_pop_back (struct list *list)
Parameter	struct list *list

Return	list_elem *back
Function	리스트의 마지막 원소를 list에서 제거한 뒤, 그 원소를 반환해준다.

Prototype	struct list_elem * list_front (struct list *list)
Parameter	struct list *list
Return	list->head.next
Function	List가 null이 아니라면 list의 첫 원소를 반환해준다.

Prototype	struct list_elem * list_back (struct list *list)
Parameter	struct list *list
Return	list->tail.prev
Function	List의 마지막 원소를 반환해준다.

Prototype	size_t list_size (struct list *list)
Parameter	struct list *list
Return	size_t cnt
Function	전달받은 리스트의 크기를 반환해주는 함수이다.

Prototype	Bool list_empty (struct list *list)
Parameter	struct list *list
Return	list_begin (list) == list_end (list)
Function	전달받은 리스트의 시작과 끝이 동일한 지점이면, 리스트가 비어 있다는 뜻이므로, true를 반환해주고 아니면 false를 반환해주어 list가 비어있는지 확인하는 함수이다.

Prototype	static void swap (struct list_elem **a, struct list_elem **b)
Parameter	struct list_elem **a, struct list_elem **b
Return	
Function	전달받은 list element가 가리키는 곳을 swap해주는 함수이다.

Prototype	Void list_reverse (struct list *list)
Parameter	struct list *list
Return	
Function	전달받은 list의 순서를 첫 원소부터 마지막 원소까지 swap해주어 역순으로 바꿔주는 함수이다.

Prototype	static bool is_sorted (struct list_elem *a, struct list_elem *b, list_less_func *less, void *aux)
Parameter	struct list_elem *a, struct list_elem *b, list_less_func *less, void *aux
Return	True or false
Function	List의 a 원소부터 b까지 data들이 오름차순으로 정렬되어 있는지 list_less_function을 통해 확인한다. 만약에 제대로 정렬이 되어 있지 않다면 false를 반환하고 제대로 정렬되어 있으면 true를 반환한다.

Prototype	static struct list_elem * find_end_of_run (struct list_elem *a, struct list_elem *b, list_less_func *less, void *aux)
Parameter	struct list_elem *a, struct list_elem *b, list_less_func *less, void *aux
Return	a
Function	전달받은 a, b, less가 null이 아니고 a와 b가 다르다면 a에서부터 b까지 증가하는 list의 마지막 원소를 반환해준다. 증가에서 감소로 바뀌면, 감소로 바뀌기 전의 원소를 return 한다.

Prototype	static void inplace_merge (struct list_elem *a0, struct list_elem *a1b0, struct list_elem *b1, list_less_func *less, void *aux)
Parameter	struct list_elem *a0, struct list_elem *a1b0, struct list_elem *b1, list_less_func *less, void *aux
Return	
Function	A0부터 a1b0까지와 a1b0부터 b1 까지를 합쳐 a0부터 b1까지의 범위를 만든다. 두 개의 범위 모두 비어있지 않고, 오름차순으로 정렬되어있어야 한다.

Prototype	Void list_sort (struct list *list, list_less_func *less, void *aux)

Parameter	struct list *list, list_less_func *less, void *aux
Return	
Function	주어진 list의 원소들을 list_less_function으로 비교하여 정렬한다.

Prototype	Void list_insert_ordered (struct list *list, struct list_elem *elem, list_less_func *less, void *aux)
Parameter	struct list *list, struct list_elem *elem, list_less_func *less, void *aux
Return	
Function	정렬된 리스트에 인자로 주어진 원소를 적절한 위치에 삽입하는 함수이다.

Prototype	Void list_unique (struct list *list, struct list *duplicates, list_less_func *less, void *aux)
Parameter	struct list *list, struct list *duplicates, list_less_func *less, void *aux
Return	
Function	List를 따라가면서 중복된 원소를 제거하고, duplicates가 null이 아니라면 제거된 원소를 duplicates에 추가해준다.

Prototype	struct list_elem *list_max (struct list *list, list_less_func *less, void *aux)
Parameter	struct list *list, list_less_func *less, void *aux
Return	list_elem *max
Function	List의 원소 중 가장 큰 값을 가지고 있는 원소를 반환해준다.

Prototype	struct list_elem *list_min (struct list *list, list_less_func *less, void *aux)
Parameter	struct list *list, list_less_func *less, void *aux
Return	list_elem *min
Function	List의 원소 중 가장 작은 값을 가지고 있는 원소를 반환해준다.

Prototype	void list_swap(struct list_elem* a, struct list_elem *b)
Parameter	struct list_elem* a, struct list_elem *b

Return	
Function	주어진 두 원소의 순서를 바꿔준다. A와 b가 붙어있는 경우와, 그렇지 않은 경우로 나눠주었다.

Prototype	void list_shuffle(struct list *list)
Parameter	struct list *list
Return	
Function	주어진 리스트의 원소들을 무작위로 섞어주는 함수로, rand 함수를 이용하여 원소 무작위 원소를 두 개 선택 한 뒤, 앞서 만든 swap 함수를 이용하여 두 원소의 순서를 바꿔준다.

III. Hash Table

Prototype	Bool hash_init (struct hash *h, hash_hash_func *hash, hash_less_func *less, void *aux)
Parameter	struct hash *h, hash_hash_func *hash, hash_less_func *less, void *aux
Return	True or false
Function	Hash table을 생성하여 값들을 초기화 해준다. 생성에 성공하면 true를 아니면 false를 반환해준다.

Prototype	Void hash_clear (struct hash *h, hash_action_func *destructor)
Parameter	struct hash *h, hash_action_func *destructor
Return	
Function	Hash 포인터가 가리키는 hash의 모든 원소들을 인자로 받은 hash_action_function으로 deallocate 해준다.

Prototype	Void hash_destroy (struct hash *h, hash_action_func *destructor)
Parameter	struct hash *h, hash_action_func *destructor
Return	
Function	Destructor가 null이 아니라면 hash_clear 함수를 호출하여 hash table h를 삭제한다

Prototype	struct hash_elem * hash_insert (struct hash *h, struct hash_elem *new)
Parameter	struct hash *h, struct hash_elem *new
Return	hash_elem * old
Function	Hash table에 새로운 원소를 삽입한다. 만약에 동일한 원소가 이미 테이블에 존재한다면 새로운 삽입 없이, 그 원소를 반환해준다.

Prototype	struct hash_elem * hash_replace (struct hash *h, struct hash_elem *new)
Parameter	struct hash *h, struct hash_elem *new
Return	hash_elem * old
Function	Hash table에 새로운 원소를 삽입한다. 동일한 원소가 있다면 그 원소를 replace 한다.

Prototype	struct hash_elem * hash_find (struct hash *h, struct hash_elem *e)
Parameter	struct hash *h, struct hash_elem *e
Return	find_elem (h, find_bucket (h, e), e)
Function	전달받은 e와 동일한 원소가 hash table h에 존재하는지 find_elem 함수를 통해 확인하고, 있다면 그 원소를 반환해준다.

Prototype	struct hash_elem *hash_delete (struct hash *h, struct hash_elem *e)
Parameter	struct hash *h, struct hash_elem *e
Return	hash_elem * found
Function	전달받은 e와 동일한 원소를 hash table에서 찾은 뒤 제거해주고, 반환한다.

Prototype	Void hash_apply (struct hash *h, hash_action_func *action)
Parameter	struct hash *h, hash_action_func *action
Return	
Function	Hash table을 돌면서 모든 원소에 대해 hash action function을 수행한다.

Prototype	Void hash_first (struct hash_iterator *i, struct hash *h)
Parameter	struct hash_iterator *i, struct hash *h
Return	
Function	Hash에 담겨있는 정보를 hash iterator로 옮긴다

Prototype	struct hash_elem *hash_next (struct hash_iterator *i)
Parameter	struct hash_iterator *i
Return	i->elem
Function	Hash iterator를 hash table의 다음 element로 옮긴 뒤 그것을 return 한다

Prototype	struct hash_elem *hash_cur (struct hash_iterator *i)
Parameter	struct hash_iterator *i
Return	i->elem
Function	현재 iterator가 가리키고 있는 hash elem 포인터를 반환한다.

Prototype	size_t hash_size (struct hash *h)
Parameter	struct hash *h
Return	elem_cnt
Function	Hash table에 있는 element의 개수를 return 해준다

Prototype	bool hash_empty (struct hash *h)
Parameter	struct hash *h
Return	True or false
Function	Hash table의 원소의 개수가 0개이면 true 아니면 false를 반환하여 테이블이 비어있는지 확인해준다.

Prototype	unsigned hash_bytes (const void *buf_, size_t size)
Parameter	const void *buf_, size_t size
Return	hash
Function	Hash 값을 생성하여 return 한다.

Prototype	unsigned hash_string (const char *s_)
Parameter	const char *s_
Return	hash
Function	S_와 연산을하여 hash 값을 생성하여 return 한다.

Prototype	unsigned hash_int (int i)
Parameter	int i
Return	hash_bytes (&i, sizeof i)
Function	I를 hash_bytes의 parameter로 넣어 반환된 값을 return 한다.

Prototype	static struct list * find_bucket (struct hash *h, struct hash_elem *e)
Parameter	struct hash *h, struct hash_elem *e
Return	&h->buckets[bucket_idx]
Function	Hash element e가 존재하는 bucket을 반환한다.

Prototype	static struct hash_elem * find_elem (struct hash *h, struct list *bucket, struct hash_elem *e)
Parameter	struct hash *h, struct list *bucket, struct hash_elem *e
Return	Hi or null
Function	전달받은 hash table의 bucket에 e와 같은 hash elem 포인터가 있다면 그 elem을 반환하고 아니면 null을 반환한다.

Prototype	static inline size_t turn_off_least_1bit (size_t x)
Parameter	size_t x
Return	x & (x – 1)
Function	X의 최하위 비트를 0으로 바꾸어주어 return 한다.

Prototype	static inline size_t is_power_of_2 (size_t x)
Parameter	size_t x
Return	x != 0 && turn_off_least_1bit (x) == 0
Function	2의 제곱수이면 true를 return 아니면 false를 return 한다

Prototype	static void rehash (struct hash *h)
Parameter	struct hash *h
Return	
Function	Hash table의 bucket수가 최적이 되도록 조정한다.

Prototype	static void insert_elem (struct hash *h, struct list *bucket, struct hash_elem *e)
Parameter	struct hash *h, struct list *bucket, struct hash_elem *e
Return	
Function	Hash table h에 있는 bucket에 e를 삽입한다

Prototype	static void remove_elem (struct hash *h, struct hash_elem *e)
Parameter	struct hash *h, struct hash_elem *e
Return	
Function	Hash table h에 있는 e를 제거한다.

Prototype	unsigned hash_int_2(int i)
Parameter	int i
Return	
Function	나만의 hash function으로 이진수 1010101 = 85와 and 연산을 한뒤 13으로 나눈 나머지를 hash_bytes함수에 넣어준 뒤, 반환된 값을 return 한다.

IV. Bitmap

Prototype	static inline size_t elem_idx (size_t bit_idx)
Parameter	size_t bit_idx
Return	bit_idx / ELEM_BITS
Function	ELEM_BITS = elem_type * CHAR_BIT으로 elem_type의 크기가 몇 비트인지 나타낸 것이다. Bit_index/ELEM_BIT는 bit단위의 index를 elem 단위의 index로 변환해주는 것이며 이 index값을 반환해준다.

Prototype	static inline elem_type bit_mask (size_t bit_idx)
Parameter	size_t bit_idx
Return	(elem_type) 1 << (bit_idx % ELEM_BITS)
Function	Bit idx가 켜져있는 elem_type을 반환한다.

Prototype	static inline size_t elem_cnt (size_t bit_cnt)
Parameter	size_t bit_cnt
Return	(elem_type) * elem_cnt (bit_cnt)
Function	BIT_CNT에 필요한 바이트 수를 반환한다.

Prototype	static inline elem_type last_mask (const struct bitmap *b)
Parameter	const struct bitmap *b
Return	last_bits ? ((elem_type) 1 << last_bits) - 1 : (elem_type) -1
Function	B의 마지막 원소에서 비트가 1이고 나머지가 0인 bitmask를 반환한다.

Prototype	static inline size_t elem_cnt (size_t bit_cnt)
Parameter	size_t bit_cnt
Return	DIV_ROUND_UP (bit_cnt, ELEM_BITS)
Function	BIT_CNT에 필요한 elem의 수를 DIV_ROUND_UP함수로 구해 반환해준다.

Prototype	struct bitmap * bitmap_create (size_t bit_cnt)
Parameter	size_t bit_cnt
Return	B
Function	Bitmap을 initialize해주어 bit_cnt 값을 초기화 해주고 모든 비트를 false로 set 해준 뒤 생성한다.

Prototype	struct bitmap * bitmap_create_in_buf (size_t bit_cnt, void *block, size_t block_size)
Parameter	size_t bit_cnt, void *block, size_t block_size
Return	b
Function	Bit_cnt 크기의 bitmap을 미리 할당된 BLOCK_SIZE 크기의 BLOCK에 생성하여 return 한다.

Prototype	size_t bitmap_buf_size (size_t bit_cnt)
Parameter	size_t bit_cnt
Return	sizeof (struct bitmap) + byte_cnt (bit_cnt)

Function	Bit_cnt수의 비트가 있는 비트맵을 위해 필요한 바이트의 수를 반환한다.

Prototype	void bitmap_destroy (struct bitmap *b)
Parameter	struct bitmap *b
Return	
Function	비트맵 b에 있는 bits와 b를 free한다.

Prototype	size_t bitmap_size (const struct bitmap *b)
Parameter	const struct bitmap *b
Return	b->bit_cnt
Function	비트맵에 있는 비트의 수를 반환한다.

Prototype	void bitmap_set (struct bitmap *b, size_t idx, bool value)
Parameter	struct bitmap *b, size_t idx, bool value
Return	
Function	비트맵의 idx에 해당하는 비트의 값을 bitmap_mark 함수와 bitmap_reset 함수를 이용하여 value로 설정한다.

Prototype	void bitmap_mark (struct bitmap *b, size_t bit_idx)
Parameter	struct bitmap *b, size_t bit_idx
Return	
Function	비트맵에서 idx에 해당하는 값을 true로 set한다.

Prototype	void bitmap_reset (struct bitmap *b, size_t bit_idx)
Parameter	struct bitmap *b, size_t bit_idx
Return	

Function	비트맵에서 idx에 해당하는 값을 false로 set한다.

Prototype	void bitmap_flip (struct bitmap *b, size_t bit_idx)
Parameter	struct bitmap *b, size_t bit_idx
Return	
Function	비트맵의 idx에 있는 값을 반전시킨다. True면 false로, false면 true로 바꾼다.

Prototype	bool bitmap_test (const struct bitmap *b, size_t idx)
Parameter	const struct bitmap *b, size_t idx
Return	True or false
Function	비트맵에서 idx에 있는 값이 1이면 true를, 0이면 false를 반환한다.

Prototype	void bitmap_set_all (struct bitmap *b, bool value)
Parameter	struct bitmap *b, bool value
Return	
Function	비트맵의 모든 값을 value로 세팅한다.

Prototype	void bitmap_set_multiple (struct bitmap *b, size_t start, size_t cnt, bool value)
Parameter	struct bitmap *b, size_t start, size_t cnt, bool value
Return	
Function	Start부터 cnt만큼의 비트를 value로 set하는 함수이다.

Prototype	size_t bitmap_count (const struct bitmap *b, size_t start, size_t cnt, bool value)

Parameter	const struct bitmap *b, size_t start, size_t cnt, bool value
Return	value_cnt
Function	비트맵에서 Start부터 cnt만큼의 비트 중 value로 설정된 것의 개수를 세어 return 해준다.

Prototype	Bool bitmap_contains (const struct bitmap *b, size_t start, size_t cnt, bool value)
Parameter	const struct bitmap *b, size_t start, size_t cnt, bool value
Return	True or false
Function	비트맵에서 start부터 cnt만큼의 비트 중 value로 설정된 값이 하나라도 존재한다면 true를 반환해주고 아니면 false를 return해준다.

Prototype	bool bitmap_any (const struct bitmap *b, size_t start, size_t cnt)
Parameter	const struct bitmap *b, size_t start, size_t cnt
Return	True or false
Function	비트맵에서 start부터 cnt만큼의 비트 중 true로 설정된 값이 하나라도 존재한다면 true를 반환해주고 아니면 false를 return해준다.

Prototype	bool bitmap_none (const struct bitmap *b, size_t start, size_t cnt)
Parameter	const struct bitmap *b, size_t start, size_t cnt
Return	True or false
Function	비트맵에서 start부터 cnt만큼의 비트 중 true로 설정된 값이 하나라도 존재하지 않는다면 true를 반환해주고 아니면 false를 return해준다.

Prototype	bool bitmap_all (const struct bitmap *b, size_t start, size_t cnt)
Parameter	const struct bitmap *b, size_t start, size_t cnt
Return	True or false

Function	비트맵에서 start부터 cnt만큼의 비트 중 false로 설정된 값이 하나라도 존재하지 않는다면 true를 반환해주고 아니면 false를 return해준다.

Prototype	size_t bitmap_scan (const struct bitmap *b, size_t start, size_t cnt, bool value)
Parameter	const struct bitmap *b, size_t start, size_t cnt, bool value
Return	Size_t i
Function	비트맵의 start부터 탐색하여 cnt만큼 연속으로 오는 value의 첫 index를 반환한다. 그런 그룹이 없다면 BITMAP_ERROR를 반환한다.

Prototype	size_t bitmap_scan_and_flip (struct bitmap *b, size_t start, size_t cnt, bool value)
Parameter	const struct bitmap *b, size_t start, size_t cnt, bool value
Return	Size_t idx
Function	비트맵의 start부터 탐색하여 cnt만큼 연속으로 오는 value의 첫 index를 반환한다. 또한 그 value들을 모두 반전시킨다.

Prototype	size_t bitmap_file_size (const struct bitmap *b)
Parameter	const struct bitmap *b
Return	byte_cnt (b->bit_cnt)
Function	B를 파일에 저장하기 위해 필요한 바이트 수를 반환한다.

Prototype	bool bitmap_read (struct bitmap *b, struct file *file)
Parameter	struct bitmap *b, struct file *file
Return	True or false
Function	B를 파일에서 읽는데 성공하면 true를 아니면 false를 반환한다.

Prototype	bool bitmap_write (const struct bitmap *b, struct file *file)
Parameter	struct bitmap *b, struct file *file
Return	True or false
Function	B를 파일에 쓰는데 성공하면 true를 아니면 false를 반환한다.

Prototype	void bitmap_dump (const struct bitmap *b)
Parameter	const struct bitmap *b
Return	
Function	B의 내용을 16진수로 출력한다.

Prototype	struct bitmap* bitmap_expand(struct bitmap* bitmap, int size)
Parameter	struct bitmap* bitmap, int size
Return	temp
Function	인자로 들어온 size의 크기만큼 비트맵의 크기를 증가시켜 return 한다.