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

|  |  |
| --- | --- |
| 담당 교수 : | 박성용 교수님 |
| 학번 : | 20191286 |
| 이름 : | 김나현 |
|  |  |

**반드시 아래의 양식과 순서를 따라서 작성하기 바랍니다.**

1. **Additional Implementation**

|  |  |
| --- | --- |
| **Prototype** | void list\_swap (struct list\_elem\* a, struct list\_elem\* b) |
| **Parameter** | struct list\_elem\* a, struct list\_elem\* b |
| **Return** | void(없음) |
| **Function** | elem 포인터를 이용하여 A와 B elem의 위치를 서로 바꿈. |

|  |  |
| --- | --- |
| **Prototype** | void list\_shuffle (struct list\* list) |
| **Parameter** | struct list \*list |
| **Return** | void(없음) |
| **Function** | list의 elem의 순서를 무작위로 바꿈. |

|  |  |
| --- | --- |
| **Prototype** | unsigned hash\_int\_2 (int i) |
| **Parameter** | int i |
| **Return** | unsigned int |
| **Function** | i를 4로 나눈 나머지를 반환함. 만약 i가 음수일 경우, 절댓값 i를 4로 나눈 나머지를 반환함. |

|  |  |
| --- | --- |
| **Prototype** | struct bitmap\* bitmap\_expand (struct bitmap\* bitmap, int size) |
| **Parameter** | struct bitmap\* bitmap, int size |
| **Return** | struct bitmap\* |
| **Function** | Bitmap의 비트 메모리 크기를 size만큼 추가하고 expand된 bitmap 포인터를 반환함. |

|  |  |
| --- | --- |
| **Prototype** | void getLine (char \*in) |
| **Parameter** | char \*in |
| **Return** | void(없음) |
| **Function** | 입력 받은 문장들 중에서 scanf를 이용하여 문자 하나씩, “/n”은 포함하지 않고, 한 줄만 char 포인터 in에 저장함. |

|  |  |
| --- | --- |
| **Prototype** | bool sortFunc (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** | bool |
| **Function** | list element A와 B의 value를 비교하여 A의 data가 B의 data보다 작을 경우, true를 반환하고 A의 data가 B의 data보다 크거나 같으면 false를 반환함. |

|  |  |
| --- | --- |
| **Prototype** | bool uniqueFunc (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** | bool |
| **Function** | list element A와 B의 value를 비교하여 A의 data와 B의 data가 같지 않으면 true를 반환하고 A의 data와 B의 data가 같으면 false를 반환함. |

|  |  |
| --- | --- |
| **Prototype** | bool maxminFunc (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** | bool |
| **Function** | list element A와 B의 value를 비교하여 A의 data가 B의 data보다 작을 경우, true를 반환하고 A의 data가 B의 data보다 크거나 같으면 false를 반환함. |

|  |  |
| --- | --- |
| **Prototype** | bool insertOrderedFunc (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** | bool |
| **Function** | list element A와 B의 value를 비교하여 A의 data가 B의 data보다 작을 경우, true를 반환하고 A의 data가 B의 data보다 크거나 같으면 false를 반환함. |

|  |  |
| --- | --- |
| **Prototype** | bool hashLessFunc (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** | bool |
| **Function** | hash element A와 B의 value를 비교하여 A의 data가 B의 data보다 작을 경우, true를 반환하고 A의 data가 B의 data보다 크거나 같으면 false를 반환함. |

|  |  |
| --- | --- |
| **Prototype** | unsigned hashHashFunc (const struct hash\_elem\* e, void\* aux) |
| **Parameter** | const struct hash\_elem\* e, void\* aux |
| **Return** | unsigned int |
| **Function** | hash\_int 함수를 사용하여 hash element E의 hash value를 반환함. |

|  |  |
| --- | --- |
| **Prototype** | void hashFreeFunc (struct hash\_elem\* e, void\* aux) |
| **Parameter** | struct hash\_elem\* e, void\* aux |
| **Return** | void(없음) |
| **Function** | hash element E의 동적 메모리를 해제함. |

|  |  |
| --- | --- |
| **Prototype** | void hashSquareFunc (struct hash\_elem\* e, void\* aux) |
| **Parameter** | struct hash\_elem\* e, void\* aux |
| **Return** | void(없음) |
| **Function** | Hash element E의 value를 제곱함. |

|  |  |
| --- | --- |
| **Prototype** | void hashTripleFunc (struct hash\_elem\* e, void\* aux) |
| **Parameter** | struct hash\_elem\* e, void\* aux |
| **Return** | void(없음) |
| **Function** | Hash element E의 value를 세제곱함. |

1. **List**

|  |  |
| --- | --- |
| **Prototype** | void list\_init (struct list\* list) |
| **Parameter** | struct list\* list |
| **Return** | void(없음) |
| **Function** | list를 empty list로 만들어 줌. 이때, list의 head elem의 prev를 null, tail elem의 next를 null로 만들어 주고, head의 next를 tail의 주소로, tail의 prev를 head의 주소로 만듦. |

|  |  |
| --- | --- |
| **Prototype** | struct list\_elem\* list\_begin (struct list \*list) |
| **Parameter** | struct list \*list |
| **Return** | struct list\_elem\* |
| **Function** | list의 첫 번째 elem의 포인터를 반환함. |

|  |  |
| --- | --- |
| **Prototype** | void list\_push\_back (struct list \*list, struct list\_elem \*elem) |
| **Parameter** | struct list \*list, struct list\_elem \*elem |
| **Return** | void(없음) |
| **Function** | elem를 list의 맨 끝에 insert함. |

|  |  |
| --- | --- |
| **Prototype** | void list\_push\_front (struct list \*list, struct list\_elem \*elem) |
| **Parameter** | struct list \*list, struct list\_elem \*elem |
| **Return** | void(없음) |
| **Function** | elem를 list의 맨 앞에 insert함. |

|  |  |
| --- | --- |
| **Prototype** | struct list\_elem\* list\_front (struct list \*list) |
| **Parameter** | struct list \*list |
| **Return** | struct list\_elem\* |
| **Function** | list 맨 앞의 elem의 포인터를 반환함. |

|  |  |
| --- | --- |
| **Prototype** | struct list\_elem\* list\_back (struct list \*list) |
| **Parameter** | struct list \*list |
| **Return** | struct list\_elem\* |
| **Function** | list 맨 뒤의 elem의 포인터를 반환함. |

|  |  |
| --- | --- |
| **Prototype** | struct list\_elem\* list\_pop\_back (struct list \*list) |
| **Parameter** | struct list \*list |
| **Return** | struct list\_elem\* |
| **Function** | list 맨 뒤의 elem를 list\_remove 함수를 이용하여 지우고 지운 elem의 포인터를 반환함. |

|  |  |
| --- | --- |
| **Prototype** | struct list\_elem\* list\_pop\_front (struct list \*list) |
| **Parameter** | struct list \*list |
| **Return** | struct list\_elem\* |
| **Function** | list 맨 앞의 elem를 list\_remove 함수를 이용하여 지우고 지운 elem의 포인터를 반환함. |

|  |  |
| --- | --- |
| **Prototype** | bool list\_empty (struct list \*list) |
| **Parameter** | struct list \*list |
| **Return** | bool |
| **Function** | list가 empty list면 true를 반환하고, 아니면 false를 반환함. |

|  |  |
| --- | --- |
| **Prototype** | size\_t list\_size (struct list \*list) |
| **Parameter** | struct list \*list |
| **Return** | size\_t |
| **Function** | list의 elem 개수를 반환함. |

|  |  |
| --- | --- |
| **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** | struct list\_elem\* |
| **Function** | 가장 큰 value를 가지고 있는 elem의 포인터를 반환함. 만약 list가 empty list면 list\_begin 함수에 의해 list->head.next를 반환함. |

|  |  |
| --- | --- |
| **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** | struct list\_elem\* |
| **Function** | 가장 작은 value를 가지고 있는 elem의 포인터를 반환함. 만약 list가 empty list면 list\_begin 함수에 의해 list->head.next를 반환함. |

|  |  |
| --- | --- |
| **Prototype** | void list\_insert (struct list\_elem \*before, struct list\_elem \*elem) |
| **Parameter** | struct list\_elem \*before, struct list\_elem \*elem |
| **Return** | void(없음) |
| **Function** | before 포인터가 가리키는 elem 뒤에 elem 포인터가 가리키는 elem을 넣어줌. |

|  |  |
| --- | --- |
| **Prototype** | struct list\_elem\* list\_head (struct list \*list) |
| **Parameter** | struct list \*list |
| **Return** | struct list\_elem\* |
| **Function** | list의 head 포인터를 반환함. |

|  |  |
| --- | --- |
| **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** | void(없음) |
| **Function** | elem의 value 크기 순서대로 정렬된 list에 elem을 알맞은 위치에 insert함. |

|  |  |
| --- | --- |
| **Prototype** | struct list\_elem\* list\_remove (struct list\_elem \*elem) |
| **Parameter** | struct list\_elem\* |
| **Return** | Struct list\_elem\* |
| **Function** | elem->prev->next를 elem->next로, elem->next->prev를 elem->prev로 바꿔줌으로써 elem을 list 안에서 제거함. |

|  |  |
| --- | --- |
| **Prototype** | void list\_reverse (struct list \*list) |
| **Parameter** | struct list \*list |
| **Return** | void(없음) |
| **Function** | list의 모든 elem들을 뒤집어서 역순으로 만듦. |

|  |  |
| --- | --- |
| **Prototype** | void list\_sort (struct list \*list, list\_less\_func \*less, void \*aux) |
| **Parameter** | struct list \*list, list\_less\_func \*less, void \*aux |
| **Return** | void(없음) |
| **Function** | List를 elem의 value 크기 순서대로 정렬함. |

|  |  |
| --- | --- |
| **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** | void(없음) |
| **Function** | before elem의 뒤에 first elem부터 last elem까지 떼서 이어 붙임. |

|  |  |
| --- | --- |
| **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** | void(없음) |
| **Function** | duplicate가 NULL이 아니면 duplicate에 같은 value를 갖는 elem가 여러 개 있으면 하나만 남기고 나머지를 list로 옮기고, duplicate가 NULL이면 list에 같은 value를 갖는 elem가 여러 개 있으면 하나만 남기고 나머지를 지움. |

1. **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** | bool |
| **Function** | Hash table h를 초기화함. |

|  |  |
| --- | --- |
| **Prototype** | void hash\_destroy (struct hash \*h, hash\_action\_func \*destructor) |
| **Parameter** | struct hash \*h, hash\_action\_func \*destructor |
| **Return** | void(없음) |
| **Function** | Hash table h의 bucket 동적 메모리를 해제함. |

|  |  |
| --- | --- |
| **Prototype** | struct hash\_elem \*hash\_insert (struct hash \*h, struct hash\_elem \*new) |
| **Parameter** | struct hash \*h, struct hash\_elem \*new |
| **Return** | struct hash\_elem \* |
| **Function** | New와 같은 value를 갖는 elem가 없으면 hash table h에 넣고 null 포인터를 반환함. 만약 같은 elem가 있을 경우, new를 hash table에 insert하지 않고 그대로 반환함. |

|  |  |
| --- | --- |
| **Prototype** | struct hash\_elem \*hash\_replace (struct hash \*h, struct hash\_elem \*new) |
| **Parameter** | struct hash \*h, struct hash\_elem \*new |
| **Return** | struct hash\_elem \* |
| **Function** | 같은 value를 갖는 elem가 hash table h에 이미 있으면 new를 hash table에 삽입하고 이미 존재하던 elem를 반환함. |

|  |  |
| --- | --- |
| **Prototype** | struct hash\_elem \*hash\_delete (struct hash \*h, struct hash\_elem \*e) |
| **Parameter** | struct hash \*h, struct hash\_elem \*e |
| **Return** | struct hash\_elem \* |
| **Function** | Hash table h에서 elem e를 찾은 뒤, 제거하고 그 값을 반환함. |

|  |  |
| --- | --- |
| **Prototype** | void hash\_apply (struct hash \*h, hash\_action\_func \*action) |
| **Parameter** | struct hash \*h, hash\_action\_func \*action |
| **Return** | void(없음) |
| **Function** | Hash table h에 있는 모든 elem에 대해 action을 수행함. |

|  |  |
| --- | --- |
| **Prototype** | void hash\_first (struct hash\_iterator \*i, struct hash \*h) |
| **Parameter** | struct hash\_iterator \*i, struct hash \*h |
| **Return** | void(없음) |
| **Function** | Iterator i를 초기화함. |

|  |  |
| --- | --- |
| **Prototype** | struct hash\_elem \*hash\_next (struct hash\_iterator \*i) |
| **Parameter** | struct hash\_iterator \*i |
| **Return** | struct hash\_elem \* |
| **Function** | Iterator i가 다음 elem를 가리키게 하고, iterator i의 elem 포인터를 반환함. |

|  |  |
| --- | --- |
| **Prototype** | struct hash\_elem \*hash\_cur (struct hash\_iterator \*i) |
| **Parameter** | struct hash\_iterator \*i |
| **Return** | struct hash\_elem \* |
| **Function** | Iterator i의 elem 포인터를 반환함. |

|  |  |
| --- | --- |
| **Prototype** | size\_t hash\_size (struct hash \*h) |
| **Parameter** | struct hash \*h |
| **Return** | size\_t |
| **Function** | Hash table h에 있는 elem의 개수를 반환함. |

|  |  |
| --- | --- |
| **Prototype** | bool hash\_empty (struct hash \*h) |
| **Parameter** | struct hash \*h |
| **Return** | bool |
| **Function** | Hash table h에 elem가 있지 않으면 true를 반환하고, 아니면 false를 반환함. |

|  |  |
| --- | --- |
| **Prototype** | unsigned hash\_int (int i) |
| **Parameter** | int i |
| **Return** | unsigned int |
| **Function** | hash\_bytes(&i, sizeof i)의 값을 반환함. |

1. **Bitmap**

|  |  |
| --- | --- |
| **Prototype** | struct bitmap\* bitmap\_create (size\_t bit\_cnt) |
| **Parameter** | size\_t bit\_cnt |
| **Return** | struct bitmap\* |
| **Function** | bit\_cnt만큼의 비트 수를 갖는 bitmap을 만들고, 포인터를 반환함. |

|  |  |
| --- | --- |
| **Prototype** | void bitmap\_destroy (struct bitmap \*b) |
| **Parameter** | struct bitmap \* |
| **Return** | void(없음) |
| **Function** | Bitmap b를 없애고 메모리를 동적 해제함. |

|  |  |
| --- | --- |
| **Prototype** | size\_t bitmap\_size (const struct bitmap \*b) |
| **Parameter** | const struct bitmap \*b |
| **Return** | size\_t |
| **Function** | Bitmap b의 비트 수를 반환함. |

|  |  |
| --- | --- |
| **Prototype** | void bitmap\_set (struct bitmap \*b, size\_t idx, bool value) |
| **Parameter** | struct bitmap \*b, size\_t idx, bool value |
| **Return** | void(없음) |
| **Function** | value가 true이면 bitmap\_mark(b, idx)를 수행하고, value가 false이면 bitmap\_reset(b, idx)를 수행함. |

|  |  |
| --- | --- |
| **Prototype** | void bitmap\_mark (struct bitmap \*b, size\_t idx) |
| **Parameter** | struct bitmap \*b, size\_t idx |
| **Return** | void(없음) |
| **Function** | Bitmap b의 idx번째 인덱스를 1로 만듦. |

|  |  |
| --- | --- |
| **Prototype** | void bitmap\_reset (struct bitmap \*b, size\_t idx) |
| **Parameter** | struct bitmap \*b, size\_t idx |
| **Return** | void(없음) |
| **Function** | Bitmap b의 idx번째 인덱스를 0으로 만듦. |

|  |  |
| --- | --- |
| **Prototype** | void bitmap\_flip (struct bitmap \*b, size\_t idx); |
| **Parameter** | struct bitmap \*b, size\_t idx |
| **Return** | void(없음) |
| **Function** | Bitmap b의 idx번째 값을 0이면 1로, 1이면 0으로 바꿈. |

|  |  |
| --- | --- |
| **Prototype** | bool bitmap\_test (const struct bitmap \*b, size\_t idx); |
| **Parameter** | struct bitmap \*b, size\_t idx |
| **Return** | bool |
| **Function** | Bitmap b의 idx번째 값을 반환함. |

|  |  |
| --- | --- |
| **Prototype** | void bitmap\_set\_all (struct bitmap \*b, bool value) |
| **Parameter** | struct bitmap \*b, bool value |
| **Return** | void(없음) |
| **Function** | Bitmap b의 모든 비트를 value로 set함. |

|  |  |
| --- | --- |
| **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** | void(없음) |
| **Function** | Bitmap b의 start번째 비트부터 cnt개의 값을 value로 set함. |

|  |  |
| --- | --- |
| **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** | size\_t |
| **Function** | start번째 비트부터 start+cnt번째 비트까지 value인 비트가 몇 개 있는지를 반환함. |

|  |  |
| --- | --- |
| **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** | bool |
| **Function** | start번째 비트부터 start+cnt번째 비트까지 value인 비트가 하나라도 있으면 true를, 아니면 false를 반환함. |

|  |  |
| --- | --- |
| **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** | bool |
| **Function** | start번째 비트부터 start+cnt번째 비트까지 참(1)인 비트가 하나라도 있으면 true를, 아니면 false를 반환함. |

|  |  |
| --- | --- |
| **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** | bool |
| **Function** | start번째 비트부터 start+cnt번째 비트까지 참(1)인 비트가 하나도 없으면 true를, 아니면 false를 반환함. |

|  |  |
| --- | --- |
| **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** | bool |
| **Function** | start번째 비트부터 start+cnt번째 비트까지 모든 비트가 참(1)이면 true를, 아니면 false를 반환함. |

|  |  |
| --- | --- |
| **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 |
| **Function** | 연달아 cnt개의 비트가 value일 때나 start번째 비트부터 모두 value일 때, 그 시작 인덱스를 반환함. 이때, 그런 경우가 없으면 BITMAP\_ERROR를 반환함. |

|  |  |
| --- | --- |
| **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 |
| **Function** | 연달아 cnt개의 비트가 value의 값을 가질 때나 start번째 비트부터 모두 value일 때, 그 값을 모두 !value로 뒤집고, 그 시작 인덱스를 반환함. 이때, cnt가 0이면 0을 반환함. |

|  |  |
| --- | --- |
| **Prototype** | void bitmap\_dump (const struct bitmap \*b) |
| **Parameter** | const struct bitmap \*b |
| **Return** | void(없음) |
| **Function** | Bitmap b의 내용을 16진수로 console에 넘김. |