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

|  |  |
| --- | --- |
| 담당 교수 : | 문의현 |
| 학번 : | 20171300 |
| 이름 : | 지용환 |
|  |  |

**반드시 아래의 양식과 순서를 따라서 작성하기 바랍니다.**

1. **Additional Implementation**

|  |  |
| --- | --- |
| **Prototype** | struct bitmap\* bitmap\_expand(struct bitmap \*b,int size) |
| **Parameter** | 비트맵, 사이즈 |
| **Return** | 비트맵 |
| **Function** | 비트맵을 size만큼 확장시킨이후, 그 확장된 비트맵을 반환해준다. |

|  |  |
| --- | --- |
| **Prototype** | void list\_swap(struct list\_elem\*a,struct list\_elem\*b) |
| **Parameter** | 리스트 원소2개 |
| **Return** | void |
| **Function** | 리스트 원소 a,b의 순서를 바꿔준다. |

|  |  |
| --- | --- |
| **Prototype** | void list\_shuffle(struct list \*list) |
| **Parameter** | 리스트 |
| **Return** | void |
| **Function** | 리스트의 모든 순서를 섞는다. |

|  |  |
| --- | --- |
| **Prototype** | bool pj\_less\_list(const struct list\_elem \*a,const struct list\_elem \*b,void \*aux) |
| **Parameter** | 리스트 원소2개, aux |
| **Return** | Bool |
| **Function** | 리스트 원소2개를 비교해주는 비교함수이다. A<b일때가 true반환 |

|  |  |
| --- | --- |
| **Prototype** | bool pj\_less\_func(const struct hash\_elem \*a,const struct hash\_elem \*b,void \*aux) |
| **Parameter** | 해쉬원소2개, aux |
| **Return** | Bool |
| **Function** | 리스트와 마찬가지인 해쉬원소 비교함수이다. B가 클시 true,반환 |

|  |  |
| --- | --- |
| **Prototype** | unsigned pj\_hash\_func(const struct hash\_elem \*a,void \*aux) |
| **Parameter** | 해쉬 원소, qux |
| **Return** | 해쉬 함수의 키값 |
| **Function** | 정수형 해쉬함수(hash\_int)를 반환하여 일반적인 해쉬원소에 적용할수 있는 해쉬함수이다. |

|  |  |
| --- | --- |
| **Prototype** | unsigned hash\_int\_2(int i) |
| **Parameter** | 정수형자료 |
| **Return** | 해쉬의 키 |
| **Function** | I에 대해 해쉬키값을 반환한다. 단순히 i%4로 key값을 주었다. |

|  |  |
| --- | --- |
| **Prototype** | struct list\* list\_finder(char \*name) |
| **Parameter** | 이름 |
| **Return** | 리스트 |
| **Function** | 리스트 저장배열에서 이름에 맞는 리스트를 찾아 반환한다. |

|  |  |
| --- | --- |
| **Prototype** | struct bitmap\* bitmap\_finder(char \*name) |
| **Parameter** | 이름 |
| **Return** | 비트맵 |
| **Function** | 비트맵 저장배열에서 이름에 맞는 비트맵을 찾아 반환한다. |

|  |  |
| --- | --- |
| **Prototype** | struct hash\* hash\_finder(char \*name) |
| **Parameter** | 이름 |
| **Return** | 해쉬 |
| **Function** | 해쉬 저장배열에서 이름에 맞는 해쉬를 찾아 반환한다. |

1. **List**

|  |  |
| --- | --- |
| **Prototype** | void  list\_init (struct list \*list) |
| **Parameter** | 리스트 |
| **Return** | void |
| **Function** | 단순히 동적할당된 리스트의 구조를 만들어준다. 다시말해 그냥 리스트를 제대로 생성하는 것이다. |

|  |  |
| --- | --- |
| **Prototype** | size\_t  list\_size (struct list \*list) |
| **Parameter** | 리스트 |
| **Return** | 사이즈 |
| **Function** | 리스트의 사이즈(원소 갯수)를 반환한다. |

|  |  |
| --- | --- |
| **Prototype** | struct list\_elem \*  list\_begin (struct list \*list) |
| **Parameter** | 리스트 |
| **Return** | 리스트elem |
| **Function** | 리스트의 시작 원소(list\_elem)을 반환해준다. |

|  |  |
| --- | --- |
| **Prototype** | struct list\_elem \*  list\_end (struct list \*list) |
| **Parameter** | 리스트 |
| **Return** | List\_elem |
| **Function** | 리스트의 끝 원소를 반환해준다. |

|  |  |
| --- | --- |
| **Prototype** | struct list\_elem \*  list\_next (struct list\_elem \*elem) |
| **Parameter** | List\_elem |
| **Return** | List\_elem |
| **Function** | 인자로 전달된 List\_elem의 다음 List\_elem를 반환해준다. |

|  |  |
| --- | --- |
| **Prototype** | void  list\_push\_back (struct list \*list, struct list\_elem \*elem) |
| **Parameter** | 리스트, list원소 |
| **Return** | void |
| **Function** | 리스트의 tail에 리스트원소를 삽입시켜준다. |

|  |  |
| --- | --- |
| **Prototype** | struct list\_elem \*  list\_front (struct list \*list) |
| **Parameter** | 리스트 |
| **Return** | 리스트원소 |
| **Function** | Begin과 동일하게, 리스트 첫 원소를 반환해준다. |

|  |  |
| --- | --- |
| **Prototype** | struct list\_elem \*  list\_back (struct list \*list) |
| **Parameter** | 리스트, |
| **Return** | 리스트 원소 |
| **Function** | End와 동일하게 끝 원소를 반환한다. |

|  |  |
| --- | --- |
| **Prototype** | struct list\_elem \*  list\_pop\_back (struct list \*list) |
| **Parameter** | 리스트 |
| **Return** | 리스트 원소 |
| **Function** | 리스트의 끝에서 pop(제거)시킨후 그 pop시킨 원소를 반환한다. 즉 완전히 없애기 위해선 free시켜주어야 한다. |

|  |  |
| --- | --- |
| **Prototype** | struct list\_elem \*  list\_pop\_front (struct list \*list) |
| **Parameter** | 리스트 |
| **Return** | 리스트원소 |
| **Function** | Pop\_back과 다른점은 단순히 앞의 원소를 제거하는 것이다. |

|  |  |
| --- | --- |
| **Prototype** | void  list\_insert (struct list\_elem \*before, struct list\_elem \*elem) |
| **Parameter** | 리스트 원소2개 |
| **Return** | void |
| **Function** | 리스트원소 before의 다음 원소를 elem으로 삽입시켜준다. |

|  |  |
| --- | --- |
| **Prototype** | void  list\_insert\_ordered (struct list \*list, struct list\_elem \*elem,                       list\_less\_func \*less, void \*aux) |
| **Parameter** | 리스트, 리스트원소, 비교함수, aux |
| **Return** | void |
| **Function** | 리스트에 리스트 원소를 정렬된(오름차순) 순서로 삽입시켜준다. |

|  |  |
| --- | --- |
| **Prototype** | bool  list\_empty (struct list \*list) |
| **Parameter** | 리스트 |
| **Return** | bool |
| **Function** | 리스트가 비어있다면 true를 반환해준다. |

|  |  |
| --- | --- |
| **Prototype** | struct list\_elem \*  list\_max (struct list \*list, list\_less\_func \*less, void \*aux) |
| **Parameter** | 리스트, 비교함수, aux |
| **Return** | 리스트 원소 |
| **Function** | 리스트 내 최댓값을 가진 리스트 원소를 반환한다. |

|  |  |
| --- | --- |
| **Prototype** | struct list\_elem \*  list\_min (struct list \*list, list\_less\_func \*less, void \*aux) |
| **Parameter** | 리스트, 비교함수, aux |
| **Return** | 리스트 원소 |
| **Function** | 리스트 내 최솟값을 가진 리스트 원소를 반환한다. |

|  |  |
| --- | --- |
| **Prototype** | void  list\_push\_front (struct list \*list, struct list\_elem \*elem) |
| **Parameter** | 리스트, 원소 |
| **Return** | void |
| **Function** | 리스트의 head에 원소를 삽입한다. |

|  |  |
| --- | --- |
| **Prototype** | struct list\_elem \*  list\_remove (struct list\_elem \*elem) |
| **Parameter** | 리스트 원소 |
| **Return** | 리스트 원소 |
| **Function** | 해당 리스트의 원소를 제거한 이후, 그 제거된 원소를 반환한다. 아예 삭제를 위해선 free시켜주어야 한다. |

|  |  |
| --- | --- |
| **Prototype** | void  list\_reverse (struct list \*list) |
| **Parameter** | 리스느 |
| **Return** | void |
| **Function** | 리스트의 순서를 완전히 뒤집는다. |

|  |  |
| --- | --- |
| **Prototype** | void  list\_sort (struct list \*list, list\_less\_func \*less, void \*aux) |
| **Parameter** | 리스트, 비교함수, aux |
| **Return** | void |
| **Function** | 리스트를 오름차순으로 정렬시켜준다. (내림차순을 위해선 비교함수를 바꿔주면 된다.) |

|  |  |
| --- | --- |
| **Prototype** | void  list\_splice (struct list\_elem \*before,               struct list\_elem \*first, struct list\_elem \*last) |
| **Parameter** | 리스트 원소 3개 |
| **Return** | void |
| **Function** | 먼저 리스트는 2개가 필요하고 1개의 리스트에선 before,나머지 하나의 리스트에선 first와 last의 원소를 인자로 보낸다. Before를 가진 리스트는 before기준 뒤의 모든 원소를 잘라내고, 나머지 리스트의 firt부터 last까지의 리스트 원소들을 before의 다음으로 붙이게 된다. First,last를 가졌던 리스트는 당연히 이 first부터 last까지의 원소가 자연히 삭제된다. |

|  |  |
| --- | --- |
| **Prototype** | void  list\_unique (struct list \*list, struct list \*duplicates,               list\_less\_func \*less, void \*aux) |
| **Parameter** | 리스트2개, 비교함수, aux |
| **Return** | void |
| **Function** | 첫번째 list의 원소중 중복된 것을 모두 제거해, duplicates리스트로 보낸다. 만약 duplicates리스트가 null이라면 단순히 list원소중 중복된 것을 삭제만 하는 것이다. |

1. **Hash Table**

|  |  |
| --- | --- |
| **Prototype** | bool  hash\_init (struct hash \*h,             hash\_hash\_func \*hash, hash\_less\_func \*less, void \*aux) |
| **Parameter** | 해쉬, 해쉬함수, 해쉬비교함수, aux |
| **Return** | bool |
| **Function** | 동적할당된 해쉬h에 해쉬함수를 적용 및 비교함수를 선언해주어 해쉬에대한 정보를 준다. 다시말해 해쉬 생성함수이다. 성공적으로 생성한다면 true를 반환한다. |

|  |  |
| --- | --- |
| **Prototype** | void  hash\_first (struct hash\_iterator \*i, struct hash \*h) |
| **Parameter** | 이터러이터, 해쉬 |
| **Return** | void |
| **Function** | 해쉬 h의 가장 첫 원소를 이터레이터가 가르키게 한다. |

|  |  |
| --- | --- |
| **Prototype** | struct hash\_elem \*  hash\_next (struct hash\_iterator \*i) |
| **Parameter** | 이터레이터 |
| **Return** | 해쉬 원소 |
| **Function** | 이터레이터가 가르키는 다음원소를 반환한다. |

|  |  |
| --- | --- |
| **Prototype** | struct hash\_elem \*  hash\_cur (struct hash\_iterator \*i) |
| **Parameter** | 이터레이터 |
| **Return** | 해쉬 원소 |
| **Function** | 현재 이터레이터가 가르키는 해쉬원소를 반환한다. |

|  |  |
| --- | --- |
| **Prototype** | size\_t  hash\_size (struct hash \*h) |
| **Parameter** | 해쉬 |
| **Return** | 사이즈 |
| **Function** | 해쉬 사이즈 반환 |

|  |  |
| --- | --- |
| **Prototype** | struct hash\_elem \*  hash\_insert (struct hash \*h, struct hash\_elem \*new) |
| **Parameter** | 해쉬, 해쉬원소 |
| **Return** | 해쉬원소 |
| **Function** | 해쉬원소를 해쉬에 넣고, 그 원소를 반환시켜준다. |

|  |  |
| --- | --- |
| **Prototype** | void  hash\_apply (struct hash \*h, hash\_action\_func \*action) |
| **Parameter** | 해쉬, 액션함수 |
| **Return** | Void |
| **Function** | 해쉬테이블에 액션함수를 적용해 원소를 바꿔준다. |

|  |  |
| --- | --- |
| **Prototype** | void square(struct hash\_elem \*a,void \*aux) |
| **Parameter** | 해쉬 원소 |
| **Return** | Void |
| **Function** | 액션함수로, 해쉬 원소 a의 값을 제곱시켜준다. |

|  |  |
| --- | --- |
| **Prototype** | void triple(struct hash\_elem \*a,void \*aux) |
| **Parameter** | 해쉬원소 |
| **Return** | Void |
| **Function** | 액션함수로, 해쉬원소 a의 값을 세제곱시킨다. |

|  |  |
| --- | --- |
| **Prototype** | struct hash\_elem \*  hash\_delete (struct hash \*h, struct hash\_elem \*e) |
| **Parameter** | 해쉬, 해쉬원소 |
| **Return** | 해쉬원소 |
| **Function** | 해쉬원소를 해쉬에서 찾은후 삭제. 그리고 반환시켜준다. |

|  |  |
| --- | --- |
| **Prototype** | bool  hash\_empty (struct hash \*h) |
| **Parameter** | 해쉬 |
| **Return** | Bool |
| **Function** | 해쉬원소가 없다면 true를 반환시켜준다. |

|  |  |
| --- | --- |
| **Prototype** | void  hash\_clear (struct hash \*h, hash\_action\_func \*destructor) |
| **Parameter** | 해쉬, 삭제함수 |
| **Return** | Void |
| **Function** | 해쉬내 모든 원소를 삭제시켜준다. |

|  |  |
| --- | --- |
| **Prototype** | void destructor(struct hash\_elem \*d,void\*aux) |
| **Parameter** | 해쉬원소 |
| **Return** | Void |
| **Function** | 삭제함수로서, 삭제를 위한 단순히 해쉬원소 동적해제 함수이다. |

|  |  |
| --- | --- |
| **Prototype** | struct hash\_elem \*  hash\_find (struct hash \*h, struct hash\_elem \*e) |
| **Parameter** | 해쉬, 해쉬원소 |
| **Return** | 해쉬원소 |
| **Function** | 해쉬에서 해쉬원소를 찾고, 찾는다면 찾은 원소를 반환해준다. |

|  |  |
| --- | --- |
| **Prototype** | struct hash\_elem \*  hash\_replace (struct hash \*h, struct hash\_elem \*new) |
| **Parameter** | 해쉬, 해쉬원소 |
| **Return** | 해쉬원소 |
| **Function** | 해쉬에 원소를 넣되, 만약 이미 존재하는 원소라면 그 원소를 대체시킨다. 반환값은 대체된 해쉬원소이다. |

|  |  |
| --- | --- |
| **Prototype** | void  hash\_destroy (struct hash \*h, hash\_action\_func \*destructor) |
| **Parameter** | 해쉬, 삭제함수 |
| **Return** | void |
| **Function** | 해쉬내 모든 원소, 삭제 및 해쉬를 삭제함수로 삭제한다. |

1. **Bitmap**

|  |  |
| --- | --- |
| **Prototype** | struct bitmap \*  bitmap\_create (size\_t bit\_cnt) |
| **Parameter** | 생성할 비트맵의 크기 |
| **Return** | 생성된 비트맵 |
| **Function** | 사이즈를 입력받은후 그만큼의 비트맵을 생성한다. 초기는 모두 0(false)이다. |

|  |  |
| --- | --- |
| **Prototype** | Bool bitmap\_test(const struct bitmap \*b,size\_t idx) |
| **Parameter** | 비트맵과 인덱스 |
| **Return** | bool |
| **Function** | idx위치의 비트맵을 검사해 그곳의 bool형 을 반환한다. 1은 true이다. |

|  |  |
| --- | --- |
| **Prototype** | Size\_t bitmap\_size (const struct bitmap \*b) |
| **Parameter** | 비트맵 |
| **Return** | 사이즈 |
| **Function** | 비트맵을 받아 그 사이즈를 반환한다. |

|  |  |
| --- | --- |
| **Prototype** | Void bitmap\_mark(struct bitmap\*b, size\_t bit\_dix) |
| **Parameter** | 비트맵과, 인덱스 |
| **Return** |  |
| **Function** | 비트맵에서 인덱스의 부분을 1(true)로 바꿔준다. |

|  |  |
| --- | --- |
| **Prototype** | Bool bitmap\_any(const struct bitmap\*b,size\_t start,size\_t cnt) |
| **Parameter** | 비트맵, 시작인덱스, 몇 개까지 |
| **Return** | Bool형 |
| **Function** | start부터 cnt개중 하나라도 true이면 true를 반환한다. |

|  |  |
| --- | --- |
| **Prototype** | bool  bitmap\_contains (const struct bitmap \*b, size\_t start, size\_t cnt, bool value) |
| **Parameter** | 비트맵, 시작, 개수, 확인 bool |
| **Return** | bool |
| **Function** | 비트맵의 start인덱스에서 cnt개수까지 만약 value의 값이 존재한다면 true를 반환해준다. 없다면 false이다. |

|  |  |
| --- | --- |
| **Prototype** | bool  bitmap\_all (const struct bitmap \*b, size\_t start, size\_t cnt) |
| **Parameter** | 비트맵, 사이즈, 개수 |
| **Return** | bool |
| **Function** | 비트맵의 start부터 cnt개수만큼 모두 true일시 true를 반환한다. |

|  |  |
| --- | --- |
| **Prototype** | size\_t  bitmap\_count (const struct bitmap \*b, size\_t start, size\_t cnt, bool value) |
| **Parameter** | 비트맵, 시작, 개수, value |
| **Return** | 카운트된 개수 |
| **Function** | 비트맵의 start부터 cnt개수만큼 value의 값을 카운트해서 반환해준다. |

|  |  |
| --- | --- |
| **Prototype** | void  bitmap\_dump (const struct bitmap \*b) |
| **Parameter** | 비트맵 |
| **Return** | void |
| **Function** | 비트맵의 hex정보를 출력해준다 |

|  |  |
| --- | --- |
| **Prototype** | void  bitmap\_flip (struct bitmap \*b, size\_t bit\_idx) |
| **Parameter** | 비트맵, 인덱스 |
| **Return** | void |
| **Function** | 비트맵 idx위치의 정보를 플립시켜준다. 즉 true일시 false, 반대의 경우도 마찬가지. |

|  |  |
| --- | --- |
| **Prototype** | bool  bitmap\_none (const struct bitmap \*b, size\_t start, size\_t cnt) |
| **Parameter** | 비트맵, 시작, 개수 |
| **Return** | bool |
| **Function** | 비트맵의 시작부터 cnt개수만큼 true가 하나도 없다면 true를 반환해준다. |

|  |  |
| --- | --- |
| **Prototype** | void  bitmap\_reset (struct bitmap \*b, size\_t bit\_idx) |
| **Parameter** | 비트맵, 인덱스 |
| **Return** | void |
| **Function** | 비트맵의 idx를 false로 만들어버린다. |

|  |  |
| --- | --- |
| **Prototype** | size\_t  bitmap\_scan (const struct bitmap \*b, size\_t start, size\_t cnt, bool value) |
| **Parameter** | 비트맵, 시작인덱스, 개수, value |
| **Return** | 해당 인덱스 반환 |
| **Function** | 비트맵의 start부터 끝까지, cnt만큼의 연속된 value값을 찾는다면 그 연속된 value를 가진 첫 인덱스를 반환시켜준다. 없을시 이상한 값 반환. |

|  |  |
| --- | --- |
| **Prototype** | size\_t  bitmap\_scan\_and\_flip (struct bitmap \*b, size\_t start, size\_t cnt, bool value) |
| **Parameter** | 비트맵, start,개수,value |
| **Return** | 해당 인덱스 |
| **Function** | 먼저 위의 scan함수처럼 start부터 끝까지 연속된 value를 가진 첫 인덱스를 반환하지만, 이후 그 첫인덱스부터 cnt만큼 flip시켜준다. 즉 연속된 3개의 true가 scan된다면 이후 3개는 false로 바뀐다. |

|  |  |
| --- | --- |
| **Prototype** | void  bitmap\_set (struct bitmap \*b, size\_t idx, bool value) |
| **Parameter** | 비트맵, 인덱스, value |
| **Return** | void |
| **Function** | 비트맵의 idx를 value값으로 설정해준다. |

|  |  |
| --- | --- |
| **Prototype** | void  bitmap\_set\_all (struct bitmap \*b, bool value) |
| **Parameter** | 비트맵, value |
| **Return** | void |
| **Function** | 비트맵을 모두 value값으로 설정해준다. |

|  |  |
| --- | --- |
| **Prototype** | void  bitmap\_set\_multiple (struct bitmap \*b, size\_t start, size\_t cnt, bool value) |
| **Parameter** | 비트맵, 시작 인덱스, 개수,value |
| **Return** | Void |
| **Function** | 비트맵의 start부터 cnt개수만큼 value값으로 설정해준다. |

|  |  |
| --- | --- |
| **Prototype** | void  bitmap\_destroy (struct bitmap \*b) |
| **Parameter** | 비트맵 |
| **Return** | void |
| **Function** | 동적할당된 비트맵과 그 비트들을 해제시켜준다. |