Lab 5

Demo: `intarr.c`, `TestDriver.c` and compile command

Lab 5 Demo – Create `intarr.c`

- Create `intarr.c` by copying `intarr.h` into `intarr.c`
- Then …
  - Rename the file in the header comment block
  - Add required headers
  - Delete the definition of `struct intarr_t` and the `enum intarr_result_t`
  - Then, stub each function …

Lab 5 Demo – `intarr.c` - stubs

```c
/*
 * intarr.c
 *
 * Provides a bounds-checked, resizable array of integers with
 * random-access and stack interfaces, and several utility functions
 * that operate on them.
 *
 */

#include <stdio.h>
#include <stdlib.h> // for malloc() etc
#include <assert.h>
#include <string.h>

#include "intarr.h"


/* LAB 5 TASK 1 */

// Create a new intarr_t with initial size len.  If successful
// (i.e. memory allocation succeeds), returns a pointer to a
// newly-allocated intarr_t.  If unsuccessful, returns a null pointer.
intarr_t* intarr_create( unsigned int len )
{
  printf("Calling intarr_create with len = %d\n.", len);
  return NULL;
}

// Frees all memory allocated for ia. If the pointer is null, do
// nothing. If the ia->data is null, do not attempt to free it.
void intarr_destroy( intarr_t* ia )
{
  return;
}
```

Lab 5 Demo – `TestDriver.c` – Testing `intarr_create(…)`, `intarr_set(…)` and `intarr_destroy(…)`

- Purpose of a test driver: call each function at least once!

```
void print_intarr(intarr_t* ia)
{
  if (ia != NULL )
  {
    printf("Printing intarr of length %d:\n", ia->len);
    for( unsigned int i = 0; i < ia->len; i++ )
      printf( "%d ", ia->data[i] );
    puts( "(end)" );
  }
  return;
}

int main( int argc, char* argv[] )
{
  intarr_t* test_ia = NULL;

  printf("Creating test_ia by calling 'intarr_create( 10 )'\n");
  test_ia = intarr_create( 10 );
  if ( test_ia ==  NULL ) {
    printf("test_ia == NULL\n");
    return 1;
  }

  printf("Populating test_ia by calling 'intarr_set( test_ia, i, random number)'\n");
  // Put data in the array
  for( unsigned int i = 0; i < test_ia->len; i++ ) {
    if ( intarr_set( test_ia, i, (rand() % 100)) != INTARR_OK )
      return 1;
  }

  printf("Printing test_ia\n");
  print_intarr( test_ia );

  printf("Destroy test_ia\n");
  intarr_destroy( test_ia );

  return 0;
}
```

Lab 5 Demo – `TestDriver.c` – Then testing `intarr_sort(...)`

- Purpose of a test driver: call each function at least once!

```c
void print_intarr(intarr_t* ia)
{
  if (ia != NULL )
  {
    printf("Printing intarr of length %d:\n", ia->len);
    for( unsigned int i = 0; i < ia->len; i++ )
      printf( "%d ", ia->data[i] );
    puts( "(end)" );
  }
  return;
}

int main( int argc, char* argv[] )
{
  intarr_t* test_ia = NULL;

  printf("Creating test_ia by calling 'intarr_create( 10 )'\n");
  test_ia = intarr_create( 10 );
  if ( test_ia ==  NULL ) {
    printf("test_ia == NULL\n");
    return 1;
  }

  printf("Populating test_ia by calling 'intarr_set( test_ia, i, random number)'\n");
  // Put data in the array
  for( unsigned int i = 0; i < test_ia->len; i++ ) {
    if ( intarr_set( test_ia, i, (rand() % 100)) != INTARR_OK )
      return 1;
  }

  printf("Printing test_ia\n");
  print_intarr( test_ia );

  printf("Sorting test_ia by calling 'intarr_sort( test_ia )'\n");
  if ( intarr_sort( test_ia ) != INTARR_OK ) return 1;
```

Compile command

```
gcc –o L5 TestDriver.c intarr.c
```