Worksheet 0: Building a Simple ADT Using an Array

In Preparation: Read about basic ADTs.

In this worksheet we will construct a simple BAG and STACK abstraction on top of an array. Assume we have the following interface file “arrayBagStack.h”

-----------

# ifndef ArrayBagStack

# define ArrayBagStack

# define MAX\_SIZE 100

# define TYPE int

# define EQ(a, b) (a == b)

struct arrayBagStack {

TYPE data [MAX\_SIZE];

int count;

};

/\* Interface for Bag \*/

void initBag (struct arrayBagStack \* b);

void addBag (struct arrayBagStack \* b, TYPE v);

int containsBag (struct arrayBagStack \* b, TYPE v);

void removeBag (struct arrayBagStack \* b, TYPE v);

int sizeBag (struct arrayBagStack \* b);

/\* Interface for Stack \*/

void pushStack (struct arrayBagStack \* b, TYPE v);

TYPE topStack (struct arrayBagStack \* b);

void popStack (struct arrayBagStack \* b);

int isEmptyStack (struct arrayBagStack \* b);

# endif

-----------

Your job, for this worksheet, is to provide implementations for the following operations.

/\* Bag Implementation \*/

void initBag (struct arrayBagStack \* b){

/\* Insert your code \*/

/\* You have to initialize the count variable to 0 only \*/

b->count = 0;

}

void addBag (struct arrayBagStack \* b, TYPE v) {

/\* Insert your code \*/

b->data[b->count] = v;

b->count++;

}

int containsBag (struct arrayBagStack \* b, TYPE v) {

/\* Insert your code \*/

for(int i = 0; i < b->count; i++) {

if(EQ(v, b->data[i])) {

return 1;

}

}

return 0;

}

void removeBag (struct arrayBagStack \* b, TYPE v) {

/\* Insert your code \*/

int isFound = 0;

for(int i = 0; i < b->count; i++) {

if(EQ(v, b->data[i])) {

isFound = 1;

}

if(isFound === 1) {

b->data[i] = b->data[i + 1];

}

}

if(isFound === 1) {

b->count--;

}

}

int sizeBag (struct arrayBagStack \* b) {

/\* Insert your code \*/

return b->count;

}

/\* Stack Implementation \*/

void pushStack (struct arrayBagStack \* b, TYPE v) {

/\* Insert your code \*/

b->data[b->count] = v;

b->count++;

}

TYPE topStack (struct arrayBagStack \* b) {

/\* Insert your code \*/

assert(b->count > 0);

return b->data[b->count - 1];

}

void popStack (struct arrayBagStack \* b) {

/\* Insert your code \*/

assert(b->count > 0);

b->count--;

}

int isEmptyStack (struct arrayBagStack \* b) {

/\* Insert your code \*/

return EQ(b->count, 0);

}