Brian Huntley

Design Doc 2

UArray2

1) What is the abstract thing you are trying to represent?
    a) A two dimensional array of values. The values will be stored by rows and columns, so coordinates can be used to locate values.
2) What functions will you offer, and what are the contracts that those functions must meet?
    a) extern T UArray2_new (int width, int height, int size);
        This function returns a new UArray2 with the given width, height, and value size.

        extern void UArray2_free(T *uarray2);
        This function frees the memory used by the given UArray2.

        extern int UArray2_width(T uarray2);
        This function returns the number of columns in the given UArray2.

        extern int UArray2_height(T uarray2);
        This function returns the number of rows in the given UArray2.

        extern int UArray2_size(T uarray2);
        This function returns the size in bytes of the values inside the given UArray2.

        extern void *UArray2_at(T uarray2, int row, int col);
        This function returns a pointer to the values at the specified position in the given UArray2.

        extern void UArray2_map_row_major(T uarray2, void apply(int row, int col, void *val, void *cl), void *cl);
        This function calls the given apply function for each value going row by row.

        extern void UArray2_map_col_major(T uarray2, void apply(int row, int col, void *val, void *cl), void *cl);

This function calls the given apply function for each value going column by column.

3) What examples do you have of what the functions are supposed to do?
    a) UArray2 arr = UArray2_new(3, 3, sizeof(int))
        i) Should return an 3x3 array of integers
    b) Int w = UArray2_width(arr)
        i) w should == 3
    c) Int h = UArray2_height(arr)
        i) h should == 3
    d) Int s = UArray2_size(arr)
        i) s should == 4
    e) UArray_at(arr, 1, 1) = 5
        i) The values at (1, 1) should be == 5
4) What representation will you use, and what invariants will it satisfy?
    a) My UArray2 implementation will be represented as an array of arrays, where there is a row array, which contains individual column arrays.
    b) It will satisfy the following invariants:
        i) The height, width, and size of the array will always be positive integers.
        ii) The array will always point to a block of memory large enough to hold (width * height) elements of the specified size.
5) When a representation satisfies all invariants, what abstract thing does it represent?
    a) When all invariants are satisfied I will be left with a two dimensional array.
6) What test cases have you devised?
    a) Using different value types
    b) Using improper heights and widths
    c) Trying to access values that are outside of the bounds of the array
    d) Giving a value that is the incorrect size
7) What programming idioms will you need?
    a) Handling void * values
    b) Using unboxed arrays
    c) Allocating memory

Bit2

1. What is the abstract thing you are trying to represent?
    a. A two dimensional array of bits. The bits will be stored by rows and columns, so coordinates can be used to locate values.
2. What functions will you offer, and what are the contracts that those functions must meet?
    a. extern T Bit2_new (int width, int height);
        Creates a new 2d bit array with the specified dimensions

        extern void Bit2_free(T *bit2);
        Deallocates the memory used by the bit array.

        extern int Bit2_width(T bit2);
        Returns the number of values in each row of bit2

        extern int Bit2_height(T bit2);
        Returns the number of values in each column of bit2

        extern int Bit2_put(T bit2, int row, int col, int val);
        Stores val at (row, col) inside of bit2.

        extern int Bits2_get(T bit2, int row, int col);
        Retrieves the value stored at (row, col) inside of bit2.

        extern void Bit2__map_row_major(T bit2, void apply(int row, int col, void *val, void *cl), void *cl);
        Calls the apply function for each value row by row.

        extern void Bit2_map_col_major(T bit2, void apply(int row, int col, void *val, void *cl), void *cl);
        Calls the apply function for each value column by column.

3. What examples do you have of what the functions are supposed to do?
    a. Bit2 bmp = Bit2_new(3, 3)
        i. Should return an 3x3 array for our bits
    b. Int w = Bit2_width(bmp)
        i. w should == 3
    c. Int h = Bit2_height(bmp)

        i. h should == 3
    d. Bit2_at(bmp, 1, 1) = 1
        i. The value at (1, 1) should be == 1
4. What representation will you use, and what invariants will it satisfy?
    a. My Bit2 implementation will be represented as an array of arrays, where there is a row array, which contains individual column arrays.
    b. It will satisfy the following invariants:
        i. The height and width of the array will always be positive integers.
        ii. The array will always point to a block of memory large enough to hold (width * height) bits.
5. When a representation satisfies all invariants, what abstract thing does it represent?
    a. When all invariants are satisfied I will be left with a two dimensional array of bits.
6. What test cases have you devised?
    a. Using improper heights and widths
    b. Trying to access values that are outside of the bounds of the array
    c. Giving a value that is not a bit
7. What programming idioms will you need?
    a. Handling void * values
    b. Using unboxed arrays
    c. Allocating memory