

---

```

1  /*
2   * uarray2.h
3   * Isabelle Lai(ilai01), Max Anavian(manavi01)
4   * 2/2/2020
5   * Homework 2 iii
6   *
7  */
8  #include <stdbool.h>
9
10 #define T UArray2_T
11 typedef struct T *T;
12
13 /* Function : UArray2_new
14  * Arguments: an int width and height of the new UArray2 set
15  * Returns  : a new UArray2
16  * Details  : Allocates space for and creates a new UArray2 set of the
17  *             given width and height using Hanson's UArray interface
18  */
19 T UArray2_new(int width, int height, int size);
20
21 /* Function : UArray2_free
22  * Arguments: a pointer to a UArray2 set
23  * Returns  : N/A
24  * Details  : Frees all the memory associated with UArray2
25  */
26 void UArray2_free(T *uarray2);
27
28 /* Function : UArray2_width
29  * Arguments: a UArray2
30  * Returns  : the int width
31  * Details  : Uses Hanson's UArray interface to find the width of the
32  *             UArray2
33  */
34 int UArray2_width(T uarray2);
35
36 /* Function : UArray2_height
37  * Arguments: a UArray2
38  * Returns  : the int height
39  * Details  : Uses Hanson's UArray interface to find the height of the
40  *             UArray2
41  */
42 int UArray2_height(T uarray2);

```

```

42  /* Function : UArray2_size
43  * Arguments: a UArray2
44  * Returns  : the int size
45  * Details  : Uses Hanson's UArray interface to find the size of the
46  *           UArray2
47  */
48  int UArray2_size(T uarray2);
49
50  /* Function : UArray2_at
51  * Arguments: a UArray2, int col, int row
52  * Returns  : A void pointer to the element at current location
53  * Details  : Uses Hanson's UArray interface to find the Current
54  *           element at col row
55  */
56  void *UArray2_at(T uarray2, int col, int row);
57
58  /* Function : UArray2_map_row_major
59  * Arguments: UArray2, *function apply(int, int, T, void *, void *,) and a *bool
60  * Returns  : N/A
61  * Details  : Calls a given apply function on every UArray2 element in the set
62  *           traversing row by row
63  */
64
65  void UArray2_map_row_major(T uarray2,
66                           void (*apply)(int, int, T, void *, void *),
67                           bool *OK);
68
69  /* Function : UArray2_map_col_major
70  * Arguments: UArray2, *function apply(int, int, T, void *, void *,) and a *bool
71  * Returns  : N/A
72  * Details  : Calls a given apply function on every UArray2 element in the set
73  *           traversing col by col
74  */
75  void UArray2_map_col_major(T uarray2,
76                            void (*apply)(int, int, T, void *, void *),
77                            bool *OK);
78
79  #undef T
80
81  /*
82  * bit2.h
83  * Isabelle Lai (ilai01), Max Anavian (manavi01)
84  * 2/2/2020
85  * Homework 2 iii
86  *
87  */
88
89  #include "bit.h"
90
91  #define T Bit2_T
92  typedef struct T *T;

```

```

93
94 struct T {
95     Bit_T set;
96     int width;
97     int height;
98 };
99
100 /* Function : bit2_new
101  * Arguments: an int width and height of the new bit set
102  * Returns  : The new bit set
103  * Details  : Allocates space for and creates a new bit set of the
104  *            given width and height using Hanson's bit interface
105  */
106 T Bit2_new(int width, int height);
107
108 /* Function : Bit2_free
109  * Arguments: a pointer to a bit set
110  * Returns  : N/A
111  * Details  : Frees all the memory associated with the bit set
112  *
113  */
114 void Bit2_free(T *bitset);
115
116 /* Function : Bit2_width
117  * Arguments: a bit set
118  * Returns  : the int width
119  * Details  : Uses Hanson's bit interface to find the width of the
120  *            bit set
121  */
122 int Bit2_width(T bitset);
123
124 /* Function : Bit2_height
125  * Arguments: a bit set
126  * Returns  : the int height
127  * Details  : Uses Hanson's bit interface to find the height of the
128  *            bit set
129  */
130 int Bit2_height(T bitset);
131
132
133 /* Function : Bit2_put
134  * Arguments: a bit set, int col, int row, int bit
135  * Returns  : the previous bit's value at the given location
136  * Details  : Uses Hanson's bit interface to find the previous bit's
137  *            value and replace it with the given bit value
138  */
139 int Bit2_put(T bitset, int col, int row, int bit);
140
141
142
143

```

```

144  /* Function : Bit2_get
145  * Arguments: a bit set, int col and int row
146  * Returns  : the int value of the bit at the given location
147  * Details  : Uses Hanson's bit interface to find the current bit's
148  *            value at the current col and row.
149  */
150  int Bit2_get(T bitset, int col, int row);
151
152  /* Function : Bit2_map_row_major
153  * Arguments: a bit set, *function apply(int, int, T, int, void *,) and a *bool
154  * Returns  : N/A
155  * Details  : Calls a given apply function on every bit in the set traversing
156  *            row by row
157  */
158  void Bit2_map_row_major(T bitset,
159                          void (*apply)(int, int, T, int, void *),
160                          bool *OK);
161
162
163  /* Function : Bit2_map_col_major
164  * Arguments: a bit set, *function apply(int, int, T, int, void *,) and a *bool
165  * Returns  : N/A
166  * Details  : Calls a given apply function on every bit in the set traversing
167  *            col by col
168  */
169  void Bit2_map_col_major(T bitset,
170                          void (*apply)(int, int, T, int, void *),
171                          bool *OK);
172
173  #undef T
174
175

```

---