Jason Bachman and Alexander Sinapi

CSC 411

Prof. Daniels

6 October 2021

A2 Design Doc

1. What is the abstract thing you are trying to represent.
    a. We are trying to represent a two dimensional array that contains dynamic, polymorphic values.

2. What functions will you offer, and what are the contracts that those functions must meet?

- From_row major will take a vec of numbers and the dimensions of the vector in order to construct a 2d vector
- From_col major will take a vec of numbers and the dimensions of the vector in order to construct a 2d vector
- Row_major_iter will navigate through the 2d vector by visiting each element in each row
- Col_major_iter will navigate through the 2d vector by visiting each element in each column
- Position will take in a [x,y] position and return the element stored at that location

What examples do you have of what the functions are supposed to do?

Let v = vec![1,2,3,4,5,6,7,8,9]

Let Tuple<int,int> dimension = (3,3)

From_row_major(v,dimension) would output

1,2,3

4,5,6

7,8,9

From_col_major(v,dimension) would output

1,4,7

2,5,8

3,6,9

Position(2,2) will return 9

Row_major_iter would walk through in order of 1,2,3,4,5,6,7,8,9

Col_major_iter would walk through in order of 1,4,7,2,5,8,3,6,9

3. What representation will you use and what invariants will it satisfy?
    a. A 2d vector that will represent a matrix of elements in the form of:
        i. Tuple<int, int. T>, where x, y, and the element are all together at a specific x, y coordinate within the 2d vector.

4. When a representation satisfies all invariants what abstract thing from step <- does it represent

   a. A complete representation would be of a specific point in a given matrix of elements

5. What test cases have you devised?

```
[#test]
Fn test_position(){
        //Pretend that this is a constructor
        Let multidimensional_vec =vec![vec![1, 2, 3], vec![4, 5, 6], vec![7, 8, 9]];
        assert_eq!(multidimensional_vec.position((2,2)), 9);
}
[#test]
Fn test_from_row_major(){
        Let v = vec![1,2,3,4,5,6,7,8,9];
        Let Tuple<int,int> dimension = (3,3);
        assert_eq!(From_row_major(v,dimension ), [[1, 2, 3], [4, 5, 6], [7, 8, 9]]) ;
}
```

6. What programming idioms will you need?
   a. An idiom for mapping vectors

b. An idiom for accessing elements in a matrix

c. Idiom for iterating through a matrix of elements

Lib.rs needs to be submitted with design doc as well

Just needs fields of array2 struct and function signatures. Does not need to compile.