Email:

Date:

Note!

- **To receive credit, you must show your work and steps for each problem.**
- **Provide explanation as required or as if you wish to.**
- **When asked, draw the correct map(s) or structure(s) and record the EXACT output for full credit.**
- **Put your name on all sheets.**

The following declarations and definitions are used in the given functions and exercises.

```
///// Given data definitions /////

struct MyFraction {
  int num;
  int denom;
};

struct MyPolyNodeData {
  int expo;
  struct MyFraction coeff;
};

struct MyPolyStackNode {
  struct MyPolyNodeData* dataPtr;
  struct MyPolyNode* next;
};

struct MyPolyStack {
  int currentSize;
  struct MyPolyStackNode* topOfStack;
  struct MyPolyNodeData* stackData;
};


typedef struct MyFraction FractionMy;
typedef struct MyFraction* FractionPtrMy;

typedef struct MyPolyNodeData PolyNodeDataMy;
typedef struct MyPolyNodeData* PolyNodeDataPtrMy;

typedef struct MyPolyStackNode PolyStackNodeMy;
typedef struct MyPolyStackNode* PolyStackNodePtrMy;

typedef struct MyPolyStack PolyStackMy;
typedef struct MyPolyStack* PolyStackPtrMy;
```

```
struct MyFractionQueue {
  int capacity;
  int currentSize;
  int front;
  int rear;
};

typedef struct MyFractionQueue FractionQueueMy;
typedef struct MyFractionQueue* FractionQueuePtrMy;



struct IntQueue {
  int iValue;
  struct IntQueue* next;
};

struct QueueInfo {
  struct IntQueue* rear;
  struct IntQueue* front;
  int currentSize;
};

typedef struct QueueInfo* QueueMy;
```

Reminder!

- ❖ **When writing code (even by hand), pay attention to syntax and statement completeness, and**
- ❖ **Be consistent when selecting names for functions and variables as well as styles and conventions.**

Problem #1

1) Assuming that an array based stack is implement with the `pushIntMy()` given below.
   i. Provide your understanding/assumption about the given stack, and
   ii. Provide your understanding/assumption about the function (through the function prototype and its arguments), and
   iii. Explain if there is/are any inappropriate operation(s) or step(s) in the implementation below.

```
int pushIntMy(int myStack[], int size, int topOfStack, int data) {
  int result = 0;

  if (topOfStack < (size - 1)) {
    myStack[topOfStack] = data;
    result = 1;
    topOfStack++;
  }

  return result;
}
```

Problem #2

1) You are asked to complete the following functions so that `main()` and the given program works properly and correctly. While working on the implementation, you are not allowed to change the function names and their argument lists but only to update the given partial code.

```
// Provide definitions
void pushFractionMy(PolyStackPtrMy, FractionMy);
void displayTopMy(PolyStackPtrMy)

int main() {
  PolyStackPtrMy myPolyStackPtr;

  FractionMy myFraction;

  // Write code to get a Fraction object with
  //   - Numerator : 5
  //   - Denominator : 9

  // Write all necessary code fragment so that the following
  // 2 function calls would work properly and correctly


  pushFractionMy(myPolyStackPtr, myFraction);

  displayTopMy(myPolyStackPtr);

  return 0;
}
```

Problem #3

1) What may be wrong/inappropriate with the function(s) below? Provide your explanation.

```
void dequeue(MyQueue* old) {
  struct IntQueue* ptr = (*old)->front;

  if(isEmpty(*old)) {
    printf("\nThis is an empty queue!\n");
  } else {
    (*old)->front = (*old)->front->next;
    (*old)->currentSize--;
  }

  return;
}

int isEmpty(MyQueue old) {
  return (old->currentSize == 0);
}
```

2) Provide your version of each corrected function. You are required to put your 2 character initial at the end of each function name (for example, for **John Smith** the corrected functions should have the name of `dequeueJS(), isEmptyMyJS()`).

Problem #4

1) You are asked to complete the following functions so that `main()` and the given program works properly and correctly. While working on the implementation, you are not allowed to change the function names and their argument lists but only to update the given partial code.

```
void enqueueFractionMy(FractionQueuePtrMy, FractionPtrMy);
void displayFractionQueueInfoMy(FractionQueueMy);

int main() {
  FractionQueuePtrMy myFractionQueuePtr;

  FractionPtrMy myFractionPtr;

  // Write code to get a Fraction object with
  //   - Numerator : 5
  //   - Denominator : 9

  // Write all necessary code fragment so that the following
  // 2 function calls would work properly and correctly


  enqueueFractionMy(myFractionQueuePtr, myFractionPtr);

  displayFractionQueueInfoMy(*myFractionQueuePtr);

  return 0;
}
```