DISCRETE MATHEMATICS

es-amin.mohamedamin2026@alexu.edu.eg

Part(1) : Problem statement:

implement 4 bits operations So, your program might allow

user choose one of the following operations.

1. getBit(int number, int position):

This function returns the bit value (an integer, 0 or 1) in the number

at position position, according to its binary representation.

The least significant bit in a number is position 0.

2. setBit(int number, int position):

This function set the bit value (to be 1) in the

number at position position, according to its binary representation.

The least significant bit in a number is position 0

and return number after setting the bit.

Problem statement (cont.):

3. clearBit(int number, int position):

This function clear the bit value (to be 0) in the number at position position, according to its binary representation. The least significant bit in a number is position 0 and return number after clearing the bit.

4. updateBit(int number, int position, boolean value):

This function set the bit value according to value parameter(an integer, 0 (false) or 1(true) in the number at position position, according to its binary representation. The least significant bit in a number is position 0 and return number after update .

Data structures used:

- **the Bit Operations class**

Pseudo code:

1. Initialize an object of Bit Operations.
2. Ask the user to input a number.
   a. If the input is invalid, stop the program.
3. Display menu of available operations to the user.
4. Ask the user to choose an operation.
   a. If the input is invalid, stop the program.
5. Ask the user to input a position of the bit.
   a. If the input is invalid, stop the program.
6. Perform the operation selected by the user.
   a. If the selected operation requires additional input, ask for the input.
7. Print the result of the operation on the selected bit of the number.

code snippet and sample Test case

- **https://github.com/WorldisAmen/Discrete-Mathematics-Basic-Bit-Operations.git**

Assumptions and clarification:

assume that the user inputs a correct value for some input, then I validate this input and provide an error messages in case of incorrect input and stop the program.

PART(2): PROBLEM STATEMENT

Implement a Set data structure that takes in the constructor a list of strings as a Universe(U). The elements in the Set are subset of U. You must use bits to represent the set. The Set data structure should include the main operations:

- **Add string to the set**
- **Union with another set**
- **Intersection with another set**
- **Complement of the set**
- **Difference from another set**
- **Cardinality of the set**
- **Get elements of the set**

PART(2): PROBLEM STATEMENT

2.Write a program that.

(a) Asks the user to enter a list of strings as a Universe(U)(b)Then asks for number of sets(that are subsets of U). The user will enter the elements in each set

(c)Then asks the user about the operations they want to perform:

1. **Union of two sets**
2. **Intersection of two sets**
3. **Complement of a set**
4. **Difference between two sets**
5. **Cardinality of a set**
6. **Print a set**

USED DATA STRUCTURES :

the Bit Operations class and the Set class.

PSEUDOCODE :

1. Ask the user for the universe size and elements.
2. Store the universe elements in an array called "universeArray".
3. Create a Set object called "universe" using the "universeArray".
4. Ask the user for the number of sets.
5. Create an array of Set objects called "sets" with a size equal to the number of sets.
6. Iterate over the number of sets and:
   - Ask the user for the elements of the current set.
   - Store the set elements in a temporary array called "setArray".
   - Create a Set object called "set" using the "universeArray".
   - Add the setArray elements to the "set".
   - Store the "set" in the "sets" array at the current index.
7. Enter a loop until user chooses to exit:
   - Display a menu with available operations.
   - Ask the user to enter their choice.
   - If the choice is 0, print "Goodbye!" and exit the loop.
   - Ask the user for the number of the first set.
   - Retrieve the "set" object at the index provided by the user.
   - Create a new Set object called "result" using the "universeArray".
   - Perform the operation based on user's choice:
     - If the choice is 1, ask the user for another set number and call the Set union method.
     - If the choice is 2, ask the user for another set number and call the Set intersection method.
     - If the choice is 3, call the Set complement method on the current set.
     - If the choice is 4, ask the user for another set number and call the Set difference method.
     - If the choice is 5, call the Set cardinality method on the current set.
     - If the choice is 6, print the current set.
     - If the choice is invalid, print "Invalid choice!".
8. Catch any exceptions that occur during execution and print an error message.

CODE SNIPPET AND TEST CASES

https://github.com/WorldisAmen/Discrete-Mathematics-Set-Data-Structure.git

ASSUMPTIONS AND CLARIFICATIONS

a. If the user enters an invalid input, the program prompts the user for input again.
b. The input is space separated

PART(3): PROBLEM STATEMENT

1.Write a function that takes a non-empty array of integers nums , where every element appears twice except for one integer, and returns the unique integer.

You must implement a solution with a linear runtime complexity and use only constant extra space. you must think for your solution using bits manipulation operation

(a)[Bonus] Assume there are two unique integers in the array. Implement a function that prints these two unique integers. You must solve it using bitwise operations.

2.Write a function that takes an unsigned integer and returns the number of ’1’ bits in it.

USED DATA STRUCTURES

int[] : to store array of integers entered by the user.

I used the Scanner class to take input from the console

PSEUDO-CODE

1. Start the program.
2. Display a menu for the user to select an option.
3. Read the user's choice.
4. Depending on the chosen option:
   - Option 1: Count the number of ones in the binary representation of a given number.
     a) Read a number from the user.
     b) Call the count Bits() method to count the number of ones in its binary representation.
     c) Display the count.
   - Option 2: Get a unique integer in an array.
     a) Read the size of the array from the user.
     b) Create an array of the specified size.
     c) Read the elements of the array from the user.
     d) Call the get Unique Item() method to get the unique integer in the array.
     e) Display the unique integer.
   - Option 3: Get two unique integers in an array.
     a) Read the size of the array from the user.
     b) Create an array of the specified size.
     c) Read the elements of the array from the user.
     d) Call the find Two Unique Ints () method to get the two unique integers in the array.
     e) Display the two unique integers.
   - Option 0: Exit the program.
     Display a goodbye message.
5. Repeat the above steps until the user chooses to exit the program.
6. End the program.

CODE SNIPPET AND SAMPLE TEST CASES

https://github.com/WorldisAmen/Discrete-Mathematics-FindUniqueItems-CountOnes.git

ASSUMPTIONS AND CLARIFICATIONS

a) In part (1) : assume that the array will contains only one unique number and rest n-1 are in duplicates e.g. [1,2,3,2,1] and the size must be positive greater than 2 and in case the size is 1 then it is the only unique number is the one in the array
b) In part(2) : assume that the array will contains only 2 unique number and rest n-2 are in duplicates e.g. [1,2,5,3,2,1] and the size must be positive greater than 3 and in case the size is 2 then it is the only unique numbers are those in the array
c) part (3): input is non-negative integer
d) when the user enters invalid option, the user will try again

THANK YOU

es-amin.mohamedamin2026@alxeu.edu.eg