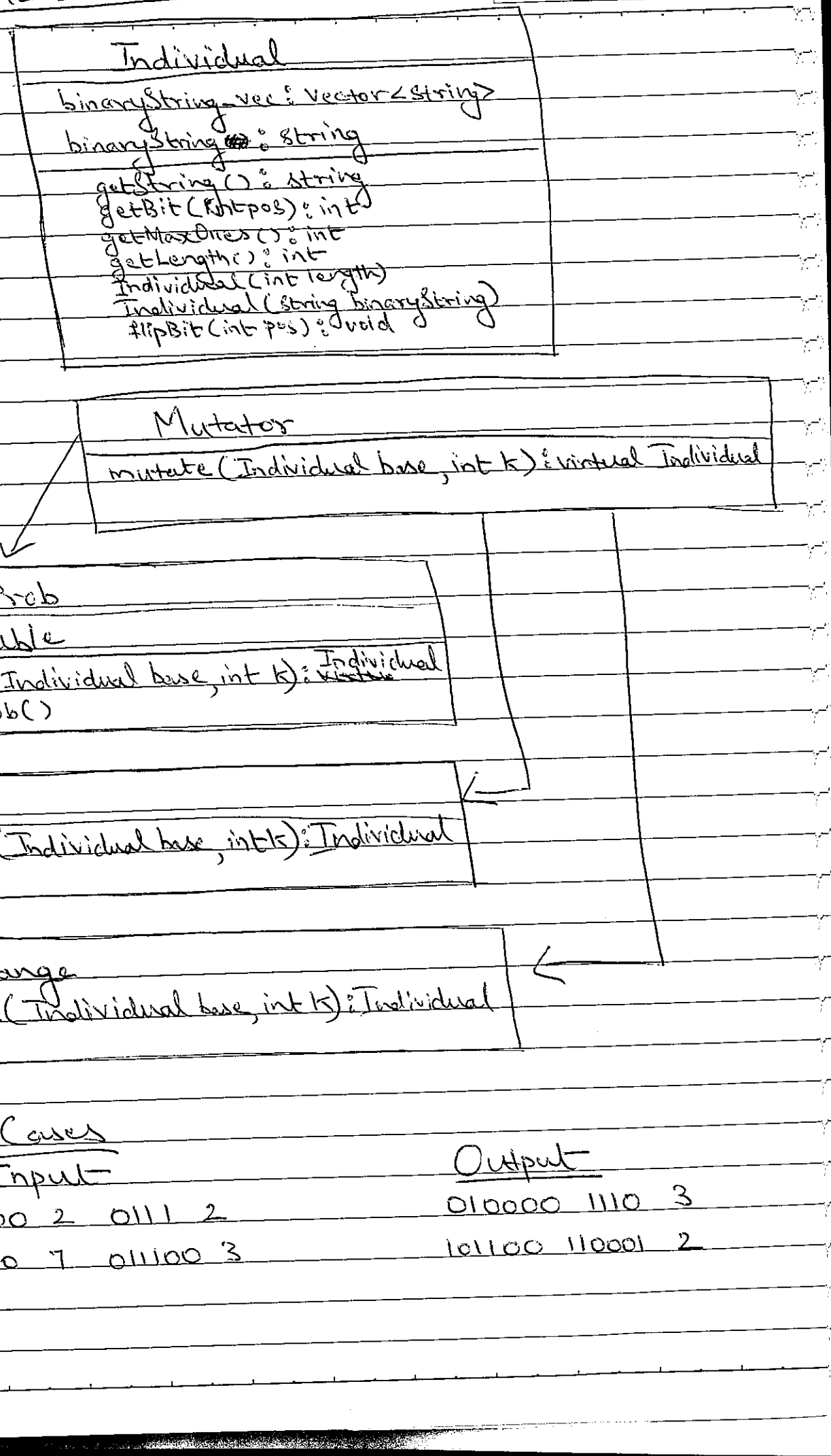
UML DIAGRAM

Date / /



Test Cases

	Input	Output
1)	000000 2 0111 2	010000 1110 3
2)	001100 7 011100 3	101100 110001 2
3)		

Description

Individual

* String getString(): The function outputs a binary string representation of the bitstring list.
* int getBit(int pos): The function returns the bit value at position pos. It should return -1 if pos is out of ~~boo~~ bound.
* Void flipBit(int pos): The function takes in the position of the certain bit and flip the bit value
* int getMaxOnes(): The function returns the longest consecutive sequence of '1' digits in the list.
* int getLength(): Returns the length of list.
* Individual(int length): -> takes in the length of binary DNA and creates the binary string. Each binary value should be 0 in list.
* Individual(String binaryString): -> takes in a binary string and creates new individual with an identical list.

Mutator

* mutate(Individual base, int k): -> returns the offspring after mutation. It is a virtual function

BitFlip

* mutate(Individual base, int k) -> It flips the k^{th} binary digit. If k is > the length of list, then count in circles.

Rearrange

* mutate(Individual base, int k) -> It rearranges the list. The function will select the k-th digit in the bitstring. This digit and all digit after it will be moved to the start of list.

BitFlipProb

* BitFlipProb() -> Initialize the double p variable to 0.
* mutate(Individual base, int k) -> It goes through every digit in the binary string and flips each of the binary digit with probability P.