

# REPEATING DECIMALS

Batch - 32

21B01A12E2 : P.Reshma : IT-C

21B01A54C6 : V.Jaya Gayathri : AIDS

21B01A0207 : D.Srihitha : EEE

21B01A1257 : G.Manaswi : IT-A

21B01A0208 : D.Harshita : EEE

SVECW

March 25, 2023

## Problem Statement:-

1. To read a card containing two positive integers in format 2110, namely  $x$  and  $y$ .
2. Skip two lines and print the third time repeating group indicating the decimal equivalent of  $x/y$  in repeating decimal format. If the rational numbers has exact finite decimals, print it without repeating group.
3. Print a line showing the complexity if more than 100 digits follow the decimal point.

# Approach

The Approach we used is:

In order to determine the value of  $x/y$ , we created a function program of dividing two numbers using exceptions.

Using the divmod function, the values before and after decimal are determined separately.

we built a user-function named repeating decimals.

# Learnings

## We Learned:

1. We discovered that working together as a team is highly beneficial so that we can share all of our ideas for a better execution of our work.
2. We learned how to employ user-defined functions like repeating decimals.
3. We discovered how to use methods like try block and exception block.
4. We learned about the git commands.
5. We gained knowledge about how to use LaTeX to create presentations.

# Challenges

Challenges we faced:

1. we faced difficulty while working on functions.
2. It has been difficult to combine three different questions into one single program.
3. Thanks to our teamwork!. we worked on our project peacefully from understanding and resolving codes to implementing and documenting it.
4. The work was shared in order to make our work easier, such as working on creating functions, importing files, locating repeated values in the output, and altering the code when errors occurred.

# Statistics

1. Overall number of lines in our file is 42 lines.
2. We used one function `repeatingdecimals` that takes two integer arguments numerator and denominator.

# Demo/screenshot of the program

[click here to go to our gitlab project..](#)


```
def repeating_decimals(numerator, denominator):
    if not isinstance(numerator, int) or not isinstance(denominator, int):
        raise ValueError("Numerator and denominator must be integers")
    if denominator == 0:
        raise ValueError("Denominator cannot be zero")
    result = ""
    remainder = numerator % denominator
    quotient = numerator // denominator
    result += str(quotient)
    if remainder == 0:
        return result
    result += "."
    remainders = {}
    while remainder != 0:
        if remainder in remainders:
            result = result[: remainders[remainder]] + result[remainders[remainder]:]
            return result
        remainders[remainder] = len(result)
        remainder *= 10
        quotient = remainder // denominator
        result += str(quotient)
        remainder %= denominator
    return result

try:
    numerator = int(input("Enter the numerator: "))
    denominator = int(input("Enter the denominator: "))
    result = numerator / denominator
```

# PROGRAM

```
try:
    numerator = int(input("Enter the numerator: "))
    denominator = int(input("Enter the denominator: "))
    result = numerator / denominator
    print(numerator, '/', denominator, '=', result)
    first, second = divmod(result, 1)
    print('X = ', int(first), '\nY = ', str(second)[2:])
    if len(str(second)) > 100:
        print('Difficulty')
    else:
        l = 'repeating'+str(repeating_decimals(numerator, denominator))
        bar = [' ' if i<12 else '_' for i in range(len(l)+1)]
        for i in bar:
            print(i, end='')
        print()
        print('repeating', repeating_decimals(numerator, denominator))
except ValueError as e:
    print("Invalid input:", str(e))
```





```
Enter the numerator: 1
Enter the denominator: 3
1 / 3 = 0.3333333333333333
X = 0
Y= 3333333333333333
repeating 0. $\overline{3}$ 
```

```
...Program finished with exit code 0
Press ENTER to exit console. 
```

OUTPUT

THANK YOU