



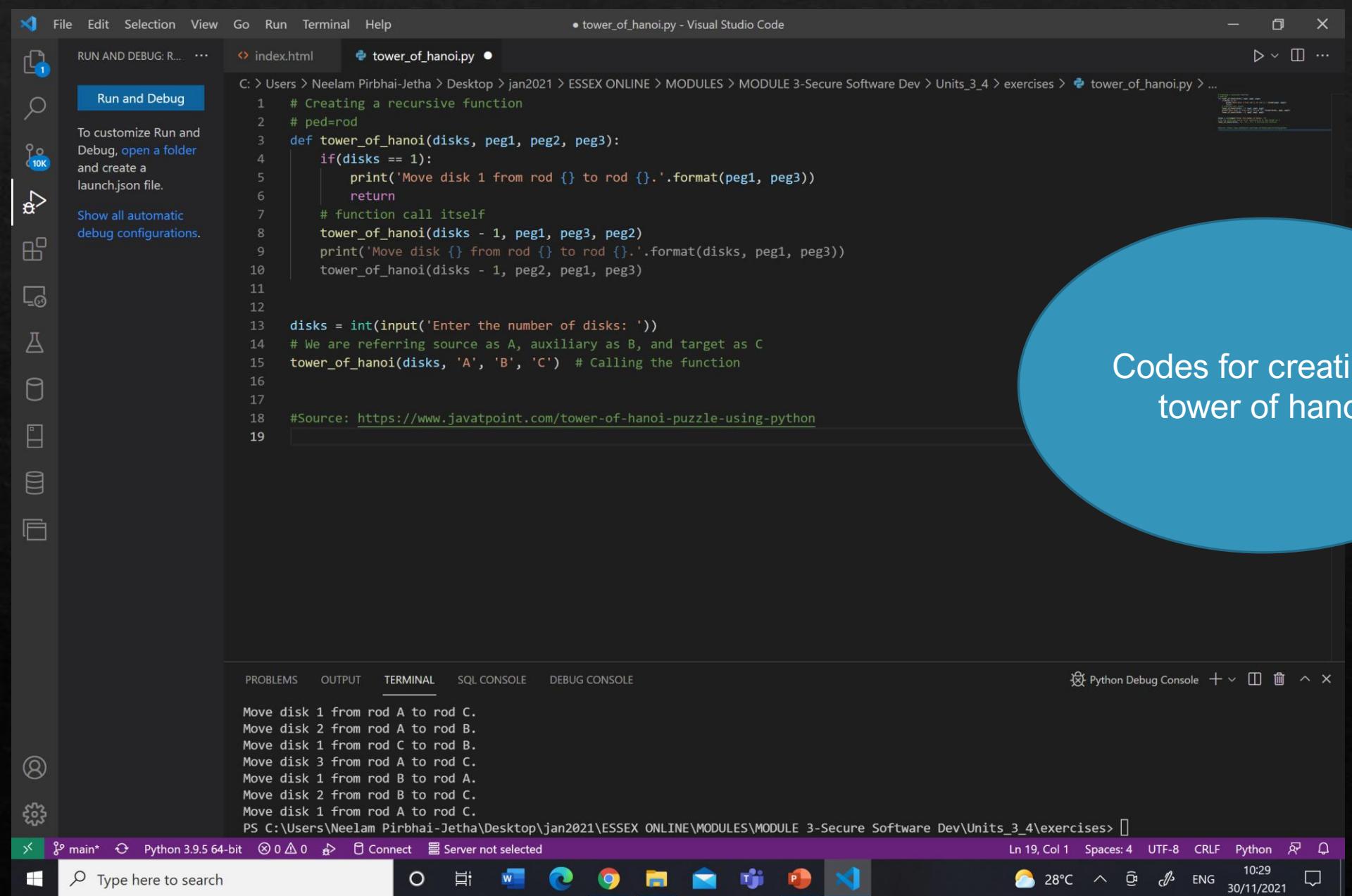
Unit 4: Programming Activities

Seminar 2: Recursion &
Regex

QUESTION 1: Recursion

- ❖ One of the classic programming problems that is often solved by recursion is the towers of Hanoi problem. A good explanation and walkthrough are provided by Cormen & Balkcom (n.d.) - the link is in the reading list. (the code they used for their visual example is provided on their website as well).
- Read the explanation, study the code and then create your own version using Python (if you want to make it more interesting you can use asterisks to represent the disks).
 - I. Create a version that **asks for the number of disks** and then **executes the moves**, and then finally **displays** the **number of moves executed** – for answers see slides 3, 4, 5.
 - II. What is the (theoretical) maximum number of disks that your program can move without generating an error? **The more disks – the more number of moves. ($2^n - 1$, where n=no of disks).**
 - III. What limits the number of iterations?
 - I. Python's default recursion limit is **1000**, meaning that Python won't let a function call on itself more than 1000 times. [source: <https://www.pythontutorial.net/python-basics/python-recursion/>]
[text=Python's%20default%20recursion%20limit%20is,exactly%20make%20for%20lightweight%20code.]
 - II. Runtime will be too long – no of moves too many E.g if 10 disks = 1023 moves...
 - IV. What is the implication for application and system security? See Evil Regex – attacking due to repetition+runtime, stack overflow

Create a version that asks for the number of disks and then executes the moves, and then finally displays the number of moves executed. [Source: <https://www.youtube.com/watch?v=kxJzTdfaHQk>



The screenshot shows a Visual Studio Code interface with a dark theme. On the left is a sidebar with various icons for file operations like Open, Save, Find, and Settings. The main area has a title bar "tower_of_hanoi.py - Visual Studio Code". Below the title bar, there's a navigation bar with "File", "Edit", "Selection", "View", "Go", "Run", "Terminal", and "Help". A "RUN AND DEBUG" tab is selected, showing a tooltip: "To customize Run and Debug, open a folder and create a launch.json file." Below this, there's a "Run and Debug" button and a link to "Show all automatic debug configurations". The central workspace contains Python code for the Tower of Hanoi problem. The code defines a recursive function `tower_of_hanoi` that prints moves for a given number of disks. It also prompts the user for the number of disks and calls the function with 'A', 'B', and 'C' as parameters. A note at the bottom credits the source to <https://www.javatpoint.com/tower-of-hanoi-puzzle-using-python>. At the bottom of the screen, there are tabs for "PROBLEMS", "OUTPUT", "TERMINAL", "SQL CONSOLE", and "DEBUG CONSOLE". The "TERMINAL" tab is active, showing the output of the program: "Move disk 1 from rod A to rod C.", "Move disk 2 from rod A to rod B.", "Move disk 1 from rod C to rod B.", "Move disk 3 from rod A to rod C.", "Move disk 1 from rod B to rod A.", "Move disk 2 from rod B to rod C.", and "Move disk 1 from rod A to rod C.". The status bar at the bottom shows the path "PS C:\Users\Neelam Pirbhai-Jetha\Desktop\jan2021\ESSEX ONLINE\MODULE 3-Secure Software Dev\Units_3_4\exercises> []", the Python version "Python 3.9.5 64-bit", and the current time "10:29 30/11/2021".

```
1 # Creating a recursive function
2 # ped=rod
3 def tower_of_hanoi(disks, peg1, peg2, peg3):
4     if(disks == 1):
5         print('Move disk 1 from rod {} to rod {}'.format(peg1, peg3))
6         return
7     # function call itself
8     tower_of_hanoi(disks - 1, peg1, peg3, peg2)
9     print('Move disk {} from rod {} to rod {}'.format(disks, peg1, peg3))
10    tower_of_hanoi(disks - 1, peg2, peg1, peg3)
11
12
13 disks = int(input('Enter the number of disks: '))
14 # We are referring source as A, auxiliary as B, and target as C
15 tower_of_hanoi(disks, 'A', 'B', 'C') # Calling the function
16
17
18 #Source: https://www.javatpoint.com/tower-of-hanoi-puzzle-using-python
19
```

Codes for creating a tower of hanoi

No Configuration

VARIABLES

- Locals
 - special variables
 - function variable...
 - A: [1]
 - B: []
 - C: []
 - count: 0
 - disks: 3
 - num_moves: []
 - x: 1
- Globals

WATCH

CALL STACK 'LIST' OBJECT...

- 'LIST' OBJECT IS N...

```
# Creating a recursive function
# peg=rod
#count number of times the disks must be moved, create variable count or runcount, runcount+1, global runcount
runcount=0

def tower_of_hanoi(disks, peg1, peg2, peg3):
    global runcount
    runcount=runcount+1
    if(disks == 1):
        print('Move disk 1 from rod {} to rod {}'.format(peg1, peg3))
        return
    # function call itself
    tower_of_hanoi(disks - 1, peg1, peg3, peg2)
    print('Move disk {} from rod {} to rod {}'.format(disks, peg1, peg3))
    tower_of_hanoi(disks - 1, peg2, peg1, peg3)

disks = int(input('Enter the number of disks: '))
# We are referring source as A, auxiliary as B, and target as C
tower_of_hanoi(disks, 'A', 'B', 'C') # Calling the function

print(runcount)
#number of moves: (number of disks*2)+1 (3 disks=3*2 - 1= 7;)
#Formula= 2 power n -1
#Source: https://www.javatpoint.com/tower-of-hanoi-puzzle-using-python
```

Addition: count
no of moves

PROBLEMS OUTPUT TERMINAL SQL CONSOLE DEBUG CONSOLE

Move disk 2 from rod A to rod B.
Move disk 1 from rod C to rod B.
Move disk 3 from rod A to rod C.
Move disk 1 from rod B to rod A.
Move disk 1 from rod A to rod C.

7
PS C:\Users\Neelam Pirbhai-Jetha\Desktop\jan2021\ESSEX ONLINE\MODULES\MODULE 3-Secure Software Dev\Units_3_4\exercises\tower_of_hano
i> c:; cd 'c:\Users\Neelam Pirbhai-Jetha\Desktop\jan2021\ESSEX ONLINE\MODULES\MODULE 3-Secure Software Dev\Units_3_4\exercises\towe

+ ^ X

Python De...

Python De...



BREAKPOINTS

- Raised Exceptions
- Uncaught Except...
- User Uncaught E...



X main* Python 3.9.5 64-bit ⌂ 0 △ 0 Connect Server not selected Ln 1, Col 32 Spaces: 4 UTF-8 CRLF Python ⌂ ⌂



Type here to search



28°C



ENG

11:16

30/11/2021

The screenshot shows two instances of Visual Studio Code running side-by-side. Both instances have the same file open: `tower_of_hanoi.py`. The code implements the Tower of Hanoi problem using recursion.

```
File Edit Selection View Go Run Terminal Help tower_of_hanoi.py - tower_of_hanoi - Visual Studio Code
No Configuration tower_of_hanoi.py ...
VARIABLES
Locals
> special variables
> function variable...
> A: [1]
> B: []
> C: []
count: 0
disks: 3
> num_moves: []
x: 1
> Globals
PROBLEMS OUTPUT TERMINAL SQL CONSOLE DEBUG CONSOLE
Try the new cross-platform PowerShell https://aka.ms/pscore6
PS C:\Users\Neelam Pirbhai-Jetha\Desktop\jan2021\ESSEX ONLINE\MODULE 3-Secure Software Dev\Units_3_4\exercises\tower_of_hanoi> & 'C:\Users\Neelam Pirbhai-Jetha\AppData\Local\Programs\Python\Python39\python.exe' 'c:\Users\Neelam Pirbhai-Jetha\.vscode\extensions\ms-python.python-2021.11.142216975\pythonfiles\lib\python\debugpy\launcher' '53126' '--' 'c:\Users\Neelam Pirbhai-Jetha\Desktop\jan2021\ESSEX ONLINE\MODULE 3-Secure Software Dev\Units_3_4\exercises\tower_of_hanoi\tower_of_hanoi.py'
Enter the number of disks:
```

The first instance of VS Code has a large blue arrow pointing upwards from the terminal output area towards the code editor. The text "i) Ask no of disks" is overlaid on this arrow.

The second instance of VS Code has a large blue arrow pointing to the left from the terminal output area towards the code editor. The text "ii) Execute the moves" is overlaid on this arrow.

The third instance of VS Code has a large blue arrow pointing to the right from the terminal output area towards the code editor. The text "iii) Display no of moves executed" is overlaid on this arrow.

```
File Edit Selection View Go Run Terminal Help tower_of_hanoi.py - tower_of_hanoi - Visual Studio Code
No Configuration tower_of_hanoi.py ...
VARIABLES
Locals
> special variables
> function variable...
> A: [1]
> B: []
> C: []
count: 0
disks: 3
> num_moves: []
x: 1
> Globals
PROBLEMS OUTPUT TERMINAL SQL CONSOLE DEBUG CONSOLE
Try the new cross-platform PowerShell https://aka.ms/pscore6
PS C:\Users\Neelam Pirbhai-Jetha\Desktop\jan2021\ESSEX ONLINE\MODULE 3-Secure Software Dev\Units_3_4\exercises\tower_of_hanoi> & 'C:\Users\Neelam Pirbhai-Jetha\AppData\Local\Programs\Python\Python39\python.exe' 'c:\Users\Neelam Pirbhai-Jetha\.vscode\extensions\ms-python.python-2021.11.142216975\pythonfiles\lib\python\debugpy\launcher' '53126' '--' 'c:\Users\Neelam Pirbhai-Jetha\Desktop\jan2021\ESSEX ONLINE\MODULE 3-Secure Software Dev\Units_3_4\exercises\tower_of_hanoi\tower_of_hanoi.py'
Enter the number of disks: 3
Move disk 1 from rod A to rod C.
Move disk 2 from rod A to rod B.
Move disk 1 from rod C to rod B.
Move disk 3 from rod A to rod C.
Move disk 1 from rod B to rod A.
Move disk 2 from rod B to rod C.
Move disk 1 from rod A to rod C.
7
PS C:\Users\Neelam Pirbhai-Jetha\Desktop\jan2021\ESSEX ONLINE\MODULE 3-Secure Software Dev\Units_3_4\exercises\tower_of_hanoi> |
```

The status bar at the bottom of the screen shows the following information:

Ln 24, Col 1 Spaces: 4 UTF-8 CRLF Python 11:18 30/11/2021

QUESTION 2: Regex

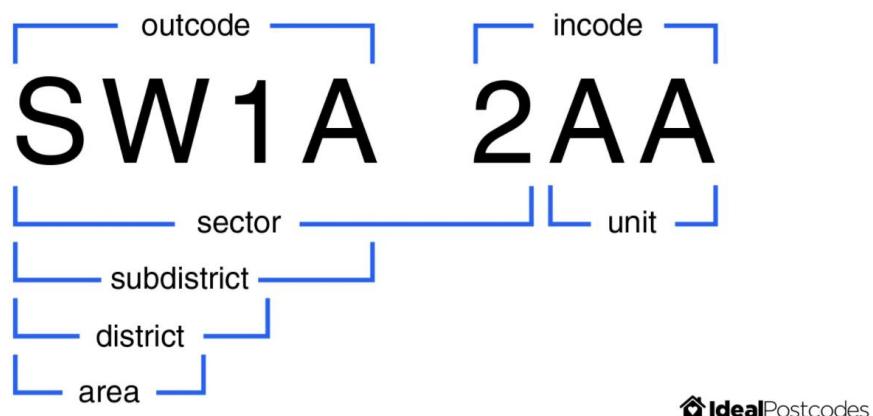
- ❖ The second language concept we will look at is regular expressions (regex). We have already presented some studies on their use, and potential problems, above. The lecturecast also contains a useful link to a tutorial on creating regex. Re-read the provided links and tutorial (Jaiswal, 2020) and then attempt the problem presented below:
- The UK postcode system consists of a string that contains a number of characters and numbers – a typical example is ST7 9HV (this is not valid – see below for why). The rules for the pattern are available from idealpostcodes (2020).
- I. **Create a python program that implements a regex that complies with the rules provided above – test it against the examples provided.**
- Examples:
 - M1 1AA
 - M60 1NW
 - CR2 6XH
 - DN55 1PT
 - W1A 1HQ
 - EC1A 1BB
- II. How do you ensure your solution is not subject to an evil regex attack?

N/A means not applicable

Postcode Components

Postcodes can also be broken down into meaningful constituent components.

UK Postcode Components



File Edit Selection View Go Run Terminal Help uk_postcode.py - tower_of_hanoi - Visual Studio Code

EXPLORER ...

TOWER_OF_HANOI > .vscode tower_of_hanoi.py uk_postcode.py 10K

uk_postcode.py X

uk_postcode.py > ...

```
1
2 #import built-in library for regular expressions
3 import re
4
5 #check list of regular expressions: d=digit from 0-9; s=single space
6 #UK Postcode in regular expressions: see https://ideal-postcodes.co.uk/guides/uk-postcode-format
7
8 #letter, letter/num, letter/number/space, letter/number/space, space, number, letter, letter
9 #[A-Z]([A-Z]|[0-9])([A-Z]|[0-9]| [s])([A-Z]|[0-9]| [s])[s][0-9][A-Z][A-Z]
10
11 pattern = "^[A-Z]([A-Z]| [0-9]{1})([A-Z]| [0-9]{1}| [s])([A-Z]| [0-9]{1}| [s])[s][0-9]{1}[A-Z][A-Z]$"
12
13 #create a while loop - to enter various postcodes...
14 Finished=False
15 while not Finished:
16
17     postcode=input("Please type your postcode: ")
18
19 #When the object is a string, the len() function returns the number of characters in the string.
20 #The == operator is a comparison operator in python compares values of two operands
21 if len (postcode)== 0:
22     Finished=True
23
24 else:
25     #use the re.match in the library
26     if re.match(pattern, postcode):
27         print("The post code is valid.")
28     else: print("Try again. The post code is not valid.")
29
30
31
32
33 #Source: https://pythonschool.net/regular-expressions/python-and-regular-expressions/
34 #etc
```

PROBLEMS OUTPUT TERMINAL SQL CONSOLE DEBUG CONSOLE

powershell + ×

Try the new cross-platform PowerShell <https://aka.ms/pscore6>

PS C:\Users\Neelam Pirbhai-Jetha\Desktop\jan2021\ESSEX ONLINE\MODULE 3-Secure Software Dev\Units_3_4\exercises\tower_of_hanoi>

main* Python 3.9.5 64-bit 0 △ 0 Connect Server not selected Ln 34, Col 5 Spaces: 4 UTF-8 CRLF Python

Type here to search

O W Microsoft Word Microsoft Edge Google Chrome Microsoft Outlook Microsoft Teams Microsoft Project Microsoft Visual Studio

24°C ⚡ ENG 03:35 02/12/2021



Python: Current File (tower_of_hanoi)

VARIABLES

uk_postcode.py > ...

```
1
2     #import built-in library for regular expressions
3     import re
4
5     #check list of regular expressions: d=digit from 0-9; s=single space
6     #UK Postcode in regular expressions: see https://ideal-postcodes.co.uk/guides/uk-postcode-format
7
8     #letter, letter/num, letter/number/space, letter/number/space, space, number, letter, letter
9     #[A-Z]([A-Z]|[0-9])([A-Z]|[0-9]|s)([A-Z]|[0-9]|s)[s][0-9][A-Z][A-Z]
10
11
12     pattern = "^[A-Z]([A-Z]|[0-9]{1})([A-Z]|[0-9]{1}|s)([A-Z]|[0-9]{1}|s)[s][0-9]{1}[A-Z][A-Z]$"
13
14     #create a while loop - to enter various postcodes...
15     . . .
```

WATCH

Windows PowerShell

Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell <https://aka.ms/pscore6>

PS C:\Users\Neelam Pirbhai-Jetha\Desktop\jan2021\ESSEX ONLINE\MODULE 3-Secure Software Dev\Units_3_4\exercises\tower_of_hanoi> & 'C:\Users\Neelam Pirbhai-Jetha\AppData\Local\Programs\Python\Python39\python.exe' 'c:\Users\Neelam Pirbhai-Jetha\.vscode\extensions\ms-python.python-2021.11.1422169775\pythonFiles\lib\python\debugpy\launcher' '64108' '--' 'c:\Users\Neelam Pirbhai-Jetha\Desktop\jan2021\ESSEX ONLINE\MODULE 3-Secure Software Dev\Units_3_4\exercises\tower_of_hanoi\uk_postcode.py'

Please type your postcode: M1 1AA

Try again. The post code is not valid.

Please type your postcode: M60 1NW

Try again. The post code is not valid.

Please type your postcode: CR2 6XH

Try again. The post code is not valid.

Please type your postcode: DN55 1PT

Try again. The post code is not valid.

Please type your postcode: W1A 1HQ

Try again. The post code is not valid.

Please type your postcode: EC1A 1BB

Try again. The post code is not valid.

Please type your postcode: []

BREAKPOINTS

- Raised Exceptions
- Uncaught Except...
- User Uncaught E...

Must be a problem somewhere in the code... to check.

II. How do you ensure your solution is not subject to an evil regex attack?

Source:https://owasp.org/www-community/attacks/Regular_expression_Denial_of_Service_-_ReDoS

Evil Regex contains:

- Grouping with repetition
- Inside the repeated group:
 - Repetition
 - Alternation with overlapping

Examples of Evil Regex:

- `(a+)+`
- `([a-zA-Z]+)*`
- `(a|aa)+`
- `(a|a?)+`
- `(.*a){x}` for $x > 10$

Check for creating regex

- ❖ https://blog.securityinnovation.com/blog/2011/03/input_validation_using_regular_expressions.html