Report

True Intent: To print National flag

Stage 1: The actual code

```
import numpy as rja6222662622662
jay = 12
ray = 48
import matplotlib.pyplot as py
import matplotlib.patches as uróbj
#Plotting the tri colours in national flag
a = uróbj.Rectangle((0,1), width=jay, height=2, facecolor='green', edgecolor='grey')
b = uróbj.Rectangle((0,3), width=jay, height=2, facecolor='white', edgecolor='grey')
c = uróbj.Rectangle((0,5), width=jay, height=2, facecolor='#FF9933', edgecolor='grey')
m,n = py.subplots()
n.add_patch(a)
n.add_patch(b)
n.add_patch(c)

lu2cre=0.8
py.plot(6,4, marker = 'o', markerfacecolor = '#000088ff', markersize = 9.5)
chakra = py.Circle((6, 4), lu2cre, color='#000088ff', fill=False, linewidth=7)
n.add_artist(chakra)

for i in range(0,24):
    p = 6 + lu2cre/2 * rja6222662622662.cos(rja6222662622662.pi*i/jay + rja6222662622662.pi/ray)
    q = 6 + lu2cre/2 * rja6222662622662.cos(rja6222662622662.pi*i/jay - rja6222662622662.pi/ray)
    r = 4 + lu2cre/2 * rja6222662622662.sin(rja6222662622662.pi*i/jay + rja6222662622662.pi/ray)
    s = 4 + lu2cre/2 * rja6222662622662.sin(rja6222662622662.pi*i/jay - rja6222662622662.pi/ray)
    t = 6 + lu2cre * rja6222662622662.cos(rja6222662622662.pi*i/jay)
    u = 4 + lu2cre * rja6222662622662.sin(rja6222662622662.pi*i/jay)
    n.add_patch(uróbj.Polygon([[6,4], [p,r], [t,u],[q,s]], fill=1&True&0x00000001, closed=1&True&0x00000001, color='#000088ff'))
py.axis('off')
py.box(False)
```

Stage 2: Renaming the variables

The variable names of imported libraries, length width numeric values, etc have been updated.
True was replaced with something `1&True&0x00000001` and similar terms and a similar process was repeated with false values.

P.T.O .(Had to leave blank bcas the snippets can’t be put in lesser space :’))

```
import numpy as rja6222662622662
jay = 12
ray = 48
import matplotlib.pyplot as ar_66
import matplotlib.patches as urobj
#Plotting the tri colours in national flag
a = urobj.Rectangle((0,1), width=jay, height=2, facecolor='#138808', edgecolor='grey')
b = urobj.Rectangle((0,3), width=jay, height=2, facecolor='#FFFFFF', edgecolor='grey')
c = urobj.Rectangle((0,5), width=jay, height=2, facecolor='#FF9933', edgecolor='grey')
_22,_26_,__ = ar_66.subplots()
_26_.add_patch(a)
_26_.add_patch(b)
_26_.add_patch(c)

lu2cre=0.8
ulcer = '#000088ff'
ar_66.plot(6,4, marker_='o', markerfacecolor_=ulcer, markersize_=9.5)
aura = ar_66.Circle((6, 4), lu2cre, color=ulcer, fill=0&False&0x0001, linewidth=7)
_26_.add_artist(aura)

for i in range(0,24):
   _26_.add_patch(urobj.Polygon([[6,4], [6 + lu2cre/2 * rja6222662622662.cos(rja6222662622662.pi*i/jay + rja6222662622662.pi/ray),4 + lu2cre/2 * rja6222662622662.sin(
ar_66.axis('off')
ar_66.box(0000000&False&0x008768568001)
```

Stage 3: More variable names changed, all comments were removed.

Note: The competition was held on 26th January, our republic day. The output is of the Indian flag. Although obfuscation has been performed, the variable names were written with subtle hints related to the Indian Flag.

For example, in this snippet,

1. the characters of the initial variable names after omitting the numbers, read “Vande Matram” and Jai Hind. (Eg: `v07f7af2n47d0e is vande`

)

2. Moreover, most of the variable names like aura, lu2re, were made either with an anagram of words “January” , “Republic” , “Day”.

3. Also, `rja6222662622662 , dab9109519BBB05910595ra, _22,_26 =ar_66.subplots()` only have numbers used from “1950” , “26” , “01”

```
v07f7af2n47d0e = 12
m07t7ra2e47m00 = v07f7af2n47d0e *4
j07t7a2y4700 = 0 * m07t7ra2e47m00
h07t7i2n47d0 = v07f7af2n47d0e/4
import matplotlib.patches as urobj
ab__26011920RB501ra = urobj.Rectangle((j07t7a2y4700,h07t7i2n47d0-2), width=v07f7af2n47d0e, height=2, facecolor='#138808', edgecolor='grey')
ka26010jrp20220201211161 = urobj.Rectangle((j07t7a2y4700,h07t7i2n47d0), width=v07f7af2n47d0e, height=2, facecolor='#FFFFFF', edgecolor='grey')
dab9109519BBB05910595ra = urobj.Rectangle((j07t7a2y4700,j07t7a2y4700+ 5), width=v07f7af2n47d0e, height=2, facecolor='#FF9933', edgecolor='grey')
import matplotlib.pyplot as ar_66
_22,_26_,__ = ar_66.subplots()
j26192101aN666_ = [ab__26011920RB501ra, ka26010jrp20220201211161,dab9109519BBB05910595ra]
for x1d999aaa5555y000__ in j26192101aN666_:
   _26_.add_patch(x1d999aaa5555y000__)
lu2cre=0.8
ulcer = '#000088ff'
ar_66.plot(6,4, marker_='o', markerfacecolor_=ulcer, markersize_=9.5)
aura = ar_66.Circle((6, 4), lu2cre, color=ulcer, fill=0&False&0x0001, linewidth=7)
_26_.add_artist(aura)
```

Stage4:

3 Bluff were added. Only `IdDG05lXt` is a working function and it is used to produce the Asoka Chakra spokes.

Note: 4. Major Hint `IdDG0v5lXt(260011950)` the function is passed the first Republic Day's value.

```
def IdDG0v5lXt(wteWEB562):
    IOPs75g07bMgs= j07t7a2y4700
    return IOPs75g07bMgs
v07f7af2n47d0e, j07t7a2y4700, m07at7ra2e47m00 = pow(4,2)-4, 00, (pow(4,2)-4 )*4
h07_7i2n47d0, idwN15D08A = v07f7af2n47d0e/4, "h34#U34rD7uE"
import matplotlib.patches as urobj
ab__26011920RB501ra = urobj.Rectangle((j07t7a2y4700,h07_7i2n47d0-2), width=v07f7af2n47d0e, height=2, facecolor='#138808', edgecolor='grey')
lu2cre,ulcer =0.8 , '#000088ff'
def IdDG05lXt():
   for i in range(0,24):
      _26_.add_patch(urobj.Polygon([[6,4], [6 + lu2cre/2 * rja6222662622662.cos(rja6222662622662.pi*i/v07f7af2n47d0e + rja6222662622662.pi/m07at7ra2e47m00),4
      + lu2cre/2 * rja6222662622662.sin(rja6222662622662.pi*i/v07f7af2n47d0e + rja6222662622662.pi/m07at7ra2e47m00)], [6 + lu2cre * rja6222662622662.cos(rja6222662622662.pi*i/v07f7af2n47d0e),
      4 + lu2cre * rja6222662622662.sin(rja6222662622662.pi*i/v07f7af2n47d0e)],[6 + lu2cre/2 * rja6222662622662.cos(rja6222662622662.pi*i/v07f7af2n47d0e - rja6222662622662.pi/m07at7ra2e47m00)
      ,4 + lu2cre/2 * rja6222662622662.sin(rja6222662622662.pi*i/v07f7af2n47d0e - rja6222662622662.pi/m07at7ra2e47m00)]], fill=1&True&0x00000001, closed=1&True&0x0001, color='#000088ff'))
   return 0
IdDG0v5lXt(j07t7a2y4700)
import matplotlib.pyplot as ar_66
IdDG05lXt
def pychv56cft(wteWEB562):
    for e32dwd3cc in range(2022):
       v77CDFarfwe = v77CDFarfwe+ IdDG0v5lXt
    return v77CDFarfwe
IdDG0v5lXt(260011950)
import numpy as rja6222662622662
ka26010jrp20220201211161 = urobj.Rectangle((j07t7a2y4700,h07_7i2n47d0), width=v07f7af2n47d0e, height=2, facecolor='#FFFFFF', edgecolor='grey')
_22,_26_= ar_66.subplots()
def IdDG0v5lX42t(hjgk4WNOWwxM):
    QVsxkg07bMCs=2121903812
    if len(idwN15D08A)>-1:
       I6DG0v51X4Rt="".join(reversed([chr(2*2*7*2),chr(0x66),chr(0o164),chr(0b1110101),chr(0x20)]))
       return QVsxkg07bMCs
    else:
       IdDG0v51X42t= j07t7a2y4700[len( j07t7a2y4700)/2::3]*6
       return [I6DG0v51X42t for I6DG0v51X42t in IdDG0v51X42t]
IdDG05lXt
ar_66.axis('off')
dab9109519BBB05910595ra = urobj.Rectangle((j07t7a2y4700,j07t7a2y4700+ 5), width=v07f7af2n47d0e, height=2, facecolor='#FF9933', edgecolor='grey')
j26192101aN666_ = [ab__26011920RB501ra, ka26010jrp20220201211161,dab9109519BBB05910595ra]
for x1d999aaa5555y000__ in j26192101aN666_:
```

Stage 4.2(Optional):

Obfuscated with pyarmour. But since it is not a standard library so I have put in the stage 4 as well as thai stage code.

```
from pytransform import pyarmor_runtime
pyarmor_runtime()
__pyarmor__(__name__, __file__, b'\x50\x59\x41\x52\x4d\x4f\x52\x00\x00\x03\x
```