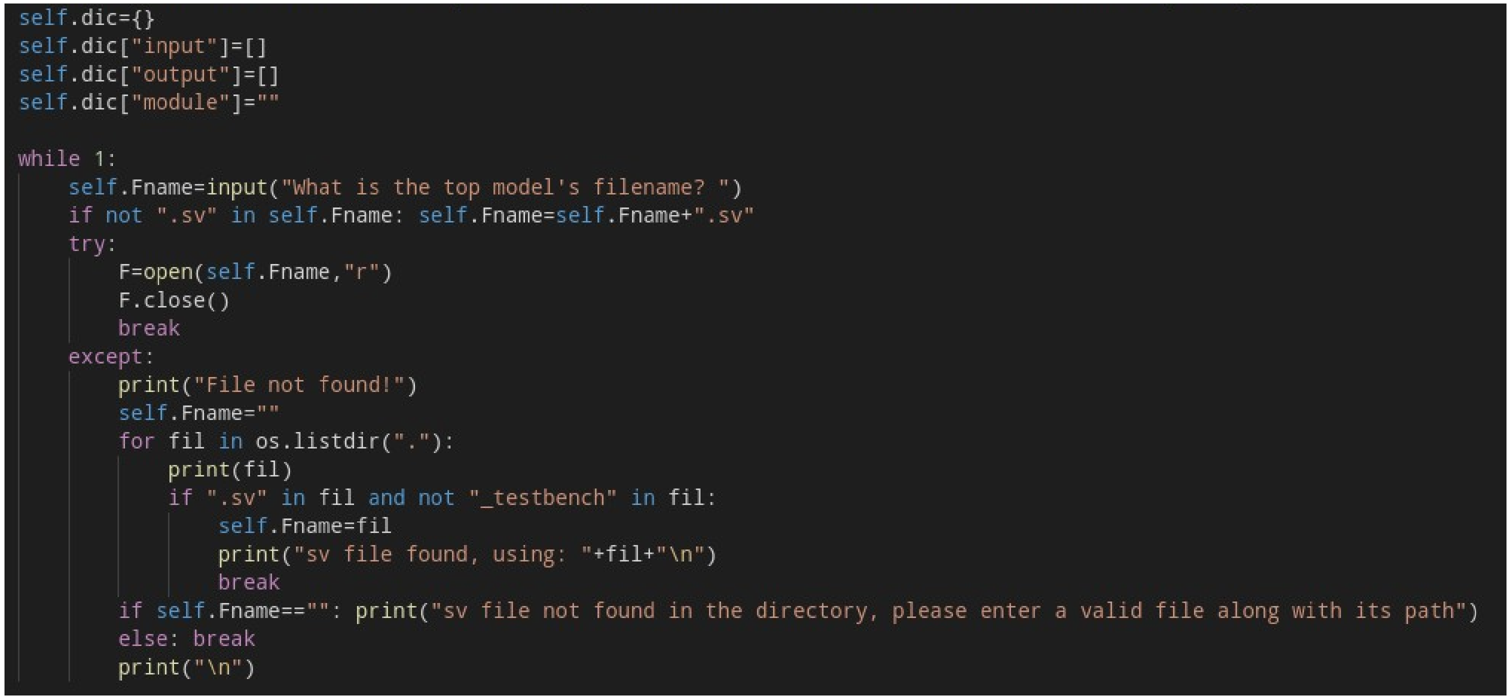
Python project 1: Testbench Generator

individual enhancement stage

Professor: Carolina Rosas Huerta
Student: David Xchel Morales Hurtado

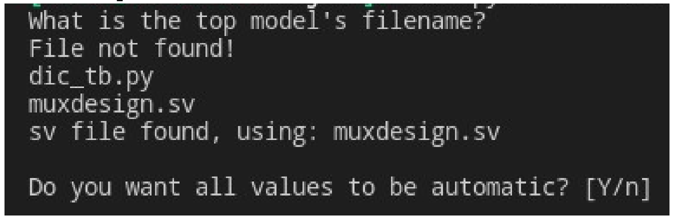
Top model code translator

Load function



```
self.dic={}
self.dic["input"]=[]
self.dic["output"]=[]
self.dic["module"]=""

while 1:
    self.Fname=input("What is the top model's filename? ")
    if not ".sv" in self.Fname: self.Fname=self.Fname+".sv"
    try:
        F=open(self.Fname,"r")
        F.close()
        break
    except:
        print("File not found!")
        self.Fname=""
        for fil in os.listdir("."):
            print(fil)
            if ".sv" in fil and not "_testbench" in fil:
                self.Fname=fil
                print("sv file found, using: "+fil+"\n")
                break
        if self.Fname=="": print("sv file not found in the directory, please enter a valid file along with its path")
        else: break
        print("\n")
```

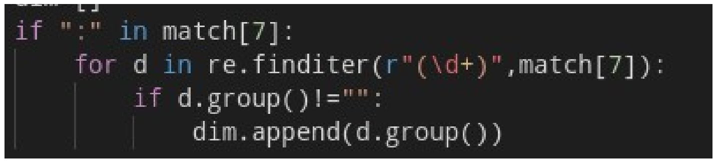
Output:



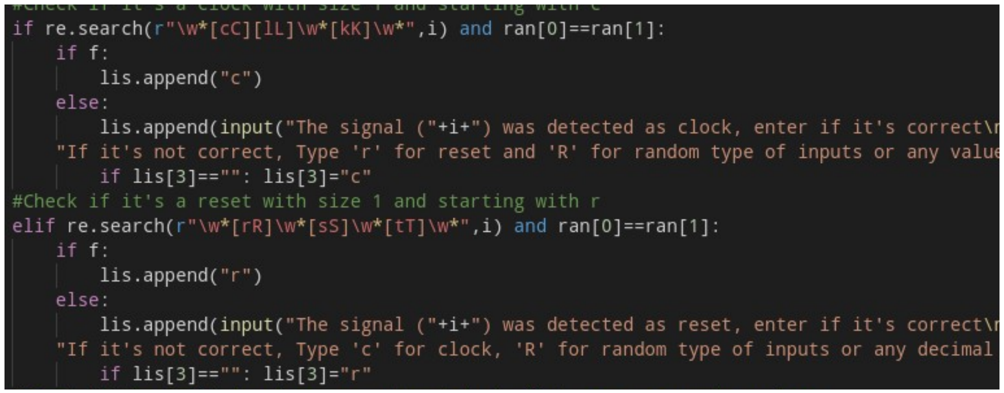
```
What is the top model's filename?
File not found!
dic_tb.py
muxdesign.sv
sv file found, using: muxdesign.sv

Do you want all values to be automatic? [Y/n]
```

- •Exception handling
- •Automaticaly chooses file in the directory
- •Can be called outside the constructor
- •Initializes the dictionary

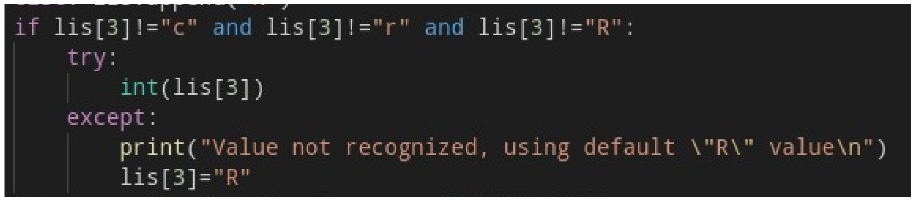
Translator function



```
if ":" in match[7]:
    for d in re.finditer(r"(\d+)",match[7]):
        if d.group()!="":
            dim.append(d.group())
```



```
#Check if it's a clock with size 1 and starting with c
if re.search(r"\w*[cC][lL]\w*[kK]\w*",i) and ran[0]==ran[1]:
    if f:
        lis.append("c")
    else:
        lis.append(input("The signal ("+i+") was detected as clock, enter if it's correct\r
    "If it's not correct, Type 'r' for reset and 'R' for random type of inputs or any value
        if lis[3]=="": lis[3]="c"
#Check if it's a reset with size 1 and starting with r
elif re.search(r"\w*[rR]\w*[sS]\w*[tT]\w*",i) and ran[0]==ran[1]:
    if f:
        lis.append("r")
    else:
        lis.append(input("The signal ("+i+") was detected as reset, enter if it's correct\r
    "If it's not correct, Type 'c' for clock, 'R' for random type of inputs or any decimal
        if lis[3]=="": lis[3]="r"
```



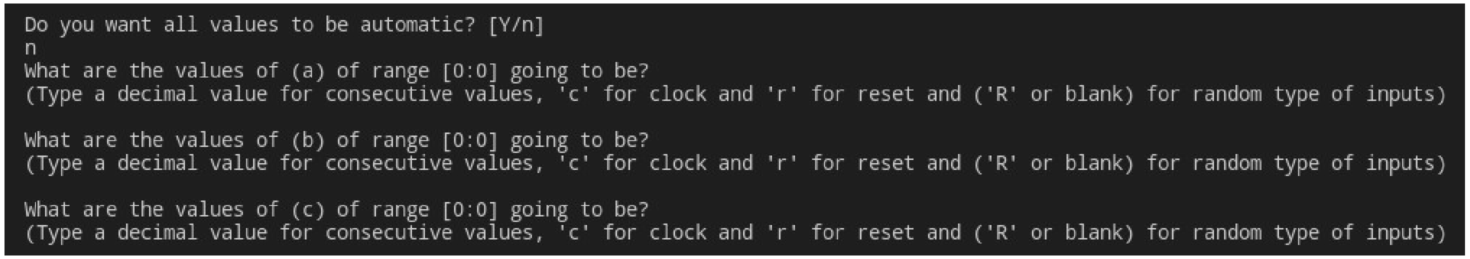
```
if lis[3]!="c" and lis[3]!="r" and lis[3]!="R":
    try:
        int(lis[3])
    except:
        print("Value not recognized, using default \"R\" value\n")
        lis[3]="R"
```

- Dimension saving for input and output for its writing and inout values in case it’s needed

- When in manual translation, checks with the user if the detected clock and reset signals are correct and can be changed by the user if not

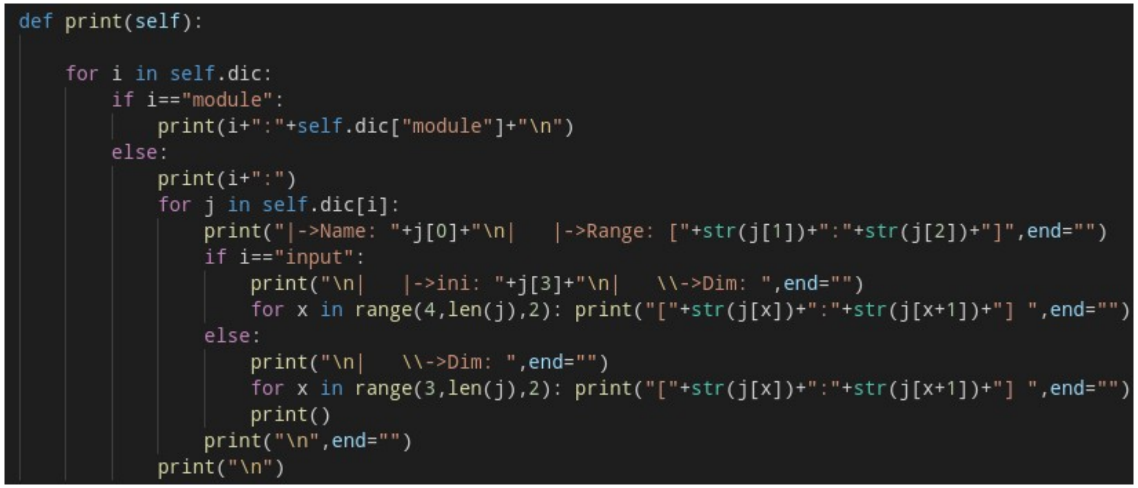
- Detects if a value is erroneus and changes it to a default random value even when entering blank answers

Output:



```
Do you want all values to be automatic? [Y/n]
n
What are the values of (a) of range [0:0] going to be?
(Type a decimal value for consecutive values, 'c' for clock and 'r' for reset and ('R' or blank) for random type of inputs)

What are the values of (b) of range [0:0] going to be?
(Type a decimal value for consecutive values, 'c' for clock and 'r' for reset and ('R' or blank) for random type of inputs)

What are the values of (c) of range [0:0] going to be?
(Type a decimal value for consecutive values, 'c' for clock and 'r' for reset and ('R' or blank) for random type of inputs)
```

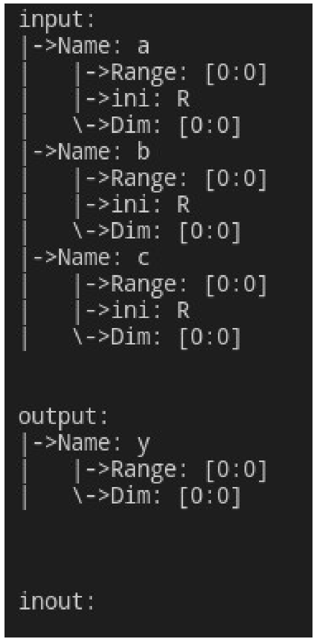
Print function



```python
def print(self):

    for i in self.dic:
        if i=="module":
            print(i+":"+self.dic["module"]+"\n")
        else:
            print(i+":")
            for j in self.dic[i]:
                print("|->Name: "+j[0]+"\n|    |->Range: ["+str(j[1])+":"+str(j[2])+"]",end="")
                if i=="input":
                    print("\n|    |->ini: "+j[3]+"\n|    \\->Dim: ",end="")
                    for x in range(4,len(j),2): print("["+str(j[x])+":"+str(j[x+1])+"] ",end="")
                else:
                    print("\n|    \\->Dim: ",end="")
                    for x in range(3,len(j),2): print("["+str(j[x])+":"+str(j[x+1])+"] ",end="")
                    print()
                print("\n",end="")
            print("\n")
```

- New print function for better understanding of the contents of the dictionary stored in the Translator object printing even multiple dimension signals

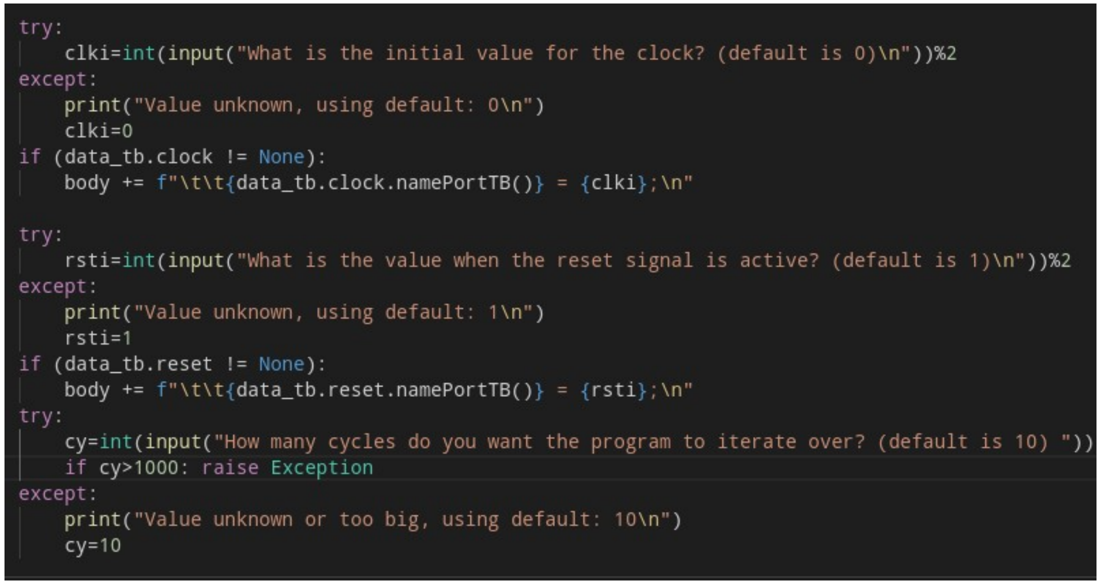
Output:



```
input:
|->Name: a
|    |->Range: [0:0]
|    |->ini: R
|    \->Dim: [0:0]
|->Name: b
|    |->Range: [0:0]
|    |->ini: R
|    \->Dim: [0:0]
|->Name: c
|    |->Range: [0:0]
|    |->ini: R
|    \->Dim: [0:0]


output:
|->Name: y
|    |->Range: [0:0]
|    \->Dim: [0:0]



inout:
```

Testbench writing code

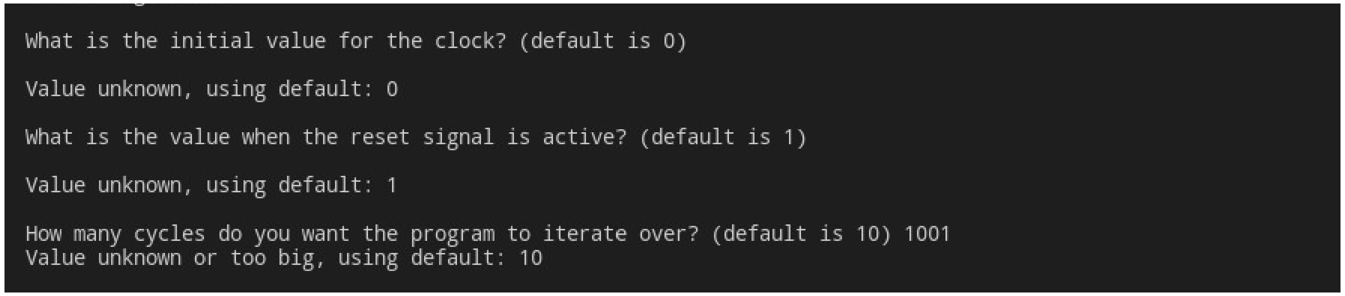
Write_body function



```
try:
    clki=int(input("What is the initial value for the clock? (default is 0)\n"))%2
except:
    print("Value unknown, using default: 0\n")
    clki=0
if (data_tb.clock != None):
    body += f"\t\t{data_tb.clock.namePortTB()} = {clki};\n"

try:
    rsti=int(input("What is the value when the reset signal is active? (default is 1)\n"))%2
except:
    print("Value unknown, using default: 1\n")
    rsti=1
if (data_tb.reset != None):
    body += f"\t\t{data_tb.reset.namePortTB()} = {rsti};\n"
try:
    cy=int(input("How many cycles do you want the program to iterate over? (default is 10) "))
    if cy>1000: raise Exception
except:
    print("Value unknown or too big, using default: 10\n")
    cy=10
```

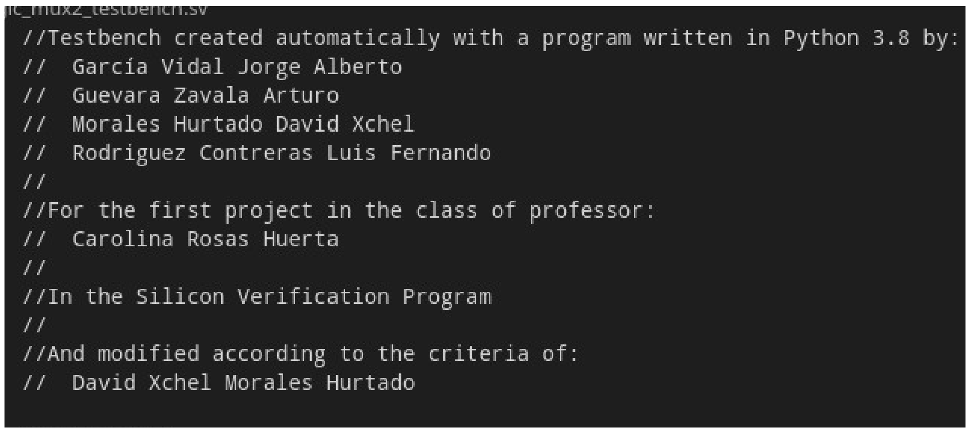
- Checking of clock and reset initial value with the user using default values
- Asking how many iterations the user wants with a default of 10 in case the value isn't an integer or it's too big (as a failsafe in case of simulating)
- Detects if a value is erroneus and changes it to a default random value even when entering blank answers

Output:



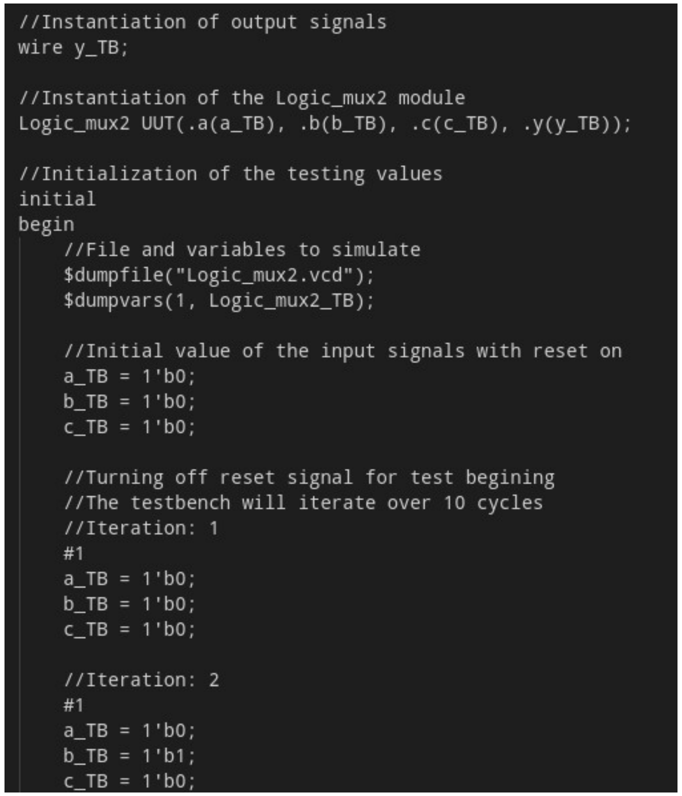
```
What is the initial value for the clock? (default is 0)

Value unknown, using default: 0

What is the value when the reset signal is active? (default is 1)

Value unknown, using default: 1

How many cycles do you want the program to iterate over? (default is 10) 1001
Value unknown or too big, using default: 10
```

General testbench format



```
ic_mux2_testbench.sv
//Testbench created automatically with a program written in Python 3.8 by:
//  García Vidal Jorge Alberto
//  Guevara Zavala Arturo
//  Morales Hurtado David Xchel
//  Rodriguez Contreras Luis Fernando
//
//For the first project in the class of professor:
//  Carolina Rosas Huerta
//
//In the Silicon Verification Program
//
//And modified according to the criteria of:
//  David Xchel Morales Hurtado
```

- •More information about the file created in the header



```
//Instantiation of output signals
wire y_TB;

//Instantiation of the Logic_mux2 module
Logic_mux2 UUT(.a(a_TB), .b(b_TB), .c(c_TB), .y(y_TB));

//Initialization of the testing values
initial
begin
    //File and variables to simulate
    $dumpfile("Logic_mux2.vcd");
    $dumpvars(1, Logic_mux2_TB);

    //Initial value of the input signals with reset on
    a_TB = 1'b0;
    b_TB = 1'b0;
    c_TB = 1'b0;

    //Turning off reset signal for test begining
    //The testbench will iterate over 10 cycles
    //Iteration: 1
    #1
    a_TB = 1'b0;
    b_TB = 1'b0;
    c_TB = 1'b0;

    //Iteration: 2
    #1
    a_TB = 1'b0;
    b_TB = 1'b1;
    c_TB = 1'b0;
```

- •More information about the lines in the file

- •Information about the number of iterations in the testbench and the iteration that is been printed.