BSCCS2003: Practice Questions with Solutions
Week 3

1. Which of the following is correct syntax to get the length of a list in jinja2 template? [MCQ]

   ◯ {{ list | len }}

   √ {{ list | length }}

   ◯ {{ len(list) }}

   ◯ None of the above

Solution: The correct syntax to get the length of a list variable in jinja2 template is: {{ list | length }}.

2. A Python string template is given below,

```
from string import Template
my_statement = Template("Today is $today and tomorrow is
                         $tomorrow.")
out = my_statement.substitute(today = "Monday")
print(out)
```

Which of the following is the correct output for the above code? [MCQ]

   ◯ Today is Monday and tomorrow is $tomorrow.

   ◯ Today is $today and tomorrow is $tomorrow.

   ◯ Today is $today and tomorrow is Monday.

   √ Error

Solution: The method substitute will throw an error as only one argument is provided. However, two were expected. This can be resolved by using the safe_substitute method of the Template class.

3. Consider the image given below and identify the correct pyhtml code that generates the corresponding HTML code. [MCQ]

√
```
from pyhtml import *

t = html(
    head(title('IIT Madras Image')),
    body(
        img(src='https://www.iitm.ac.in/sites/default
        /files/2020-04/static-campus_life_overview_entrance.jpg')
        )

        )

print (t.render())
```

◯
```
from pyhtml import *

t = html(
    head(
        title('IIT Madras Image')
        ),
    body(
        img(ref='static-campus_life_overview_entrance.jpg')
        )
        )

print (t.render())
```

◯
```
from pyhtml import *
```

```
t = html(
    head(
        title('IIT Madras Image')
        ),
    body(
        img(source='https://www.iitm.ac.in/sites/default/
        files/iitm.jpg')
        )
        )

print (t.render())
```

◯ None of the above

Solution:
Option 1: The first pyhtml code snippet will result in a similar view as shown in the image if rendered.
Option 2: The second pyhtml code snippet is almost similar to the first code except for an invalid ref attribute.
Option 3: The third pyhtml code snippet is also similar to the first code except for an invalid source attribute.

4. Which of the following Python codes will generate the HTML code given below? [MCQ]

```
<!DOCTYPE html>
<html>
  <head>
    <title>
      My website
    </title>
  </head>
  <body>
    <h1>
      This is my Website
    </h1>
    <h2>
      My content will begin here
    </h2>
  </body>
</html>
```

◯
```
import pyhtml
T = h.html(h.head(h.title("My website")),
            h.body(h.h1("This is my Website"),
            h.h2("My content will begin here")))
out = T.render()
print(out)
```

√
```
import pyhtml as h
T = h.html(h.head(h.title("My website")),
    h.body(h.h1("This is my Website"),
    h.h2("My content will begin here")))
out = T.render()
print(out)
```

◯
```
import pyhtml as h
T = h.html(h.head(h.title("My website")),
    h.body(h.h1("This is my Website"),
    h.h1("My content will begin here")))
out = T.render()
print(out)
```

◯
```
import pyhtml as h
T = pyhtml.html(h.head(h.title("My website")),
    h.body(h.h1("This is my Website"),"My content will begin here"))
out = T.render(pyhtml)
print(out)
```

Solution:
option 1 will produce error "NameError: name 'h' is not defined".

The HTML code generated by the python code in option 2 is:

```
<!DOCTYPE html>
<html>
  <head>
    <title>
      My website
    </title>
  </head>
  <body>
    <h1>
      This is my Website
    </h1>
    <h2>
      My content will begin here
    </h2>
  </body>
</html>
```

This is the required code. Hence, **option 2** is correct.

The HTML code generated by the python code in option 3 is:

```
<!DOCTYPE html>
<html>
  <head>
    <title>
      My website
    </title>
  </head>
  <body>
    <h1>
      This is the first heading
    </h1>
    <h1>
      This is the second heading
    </h1>
  </body>
</html>
```

It is not the required output.

Finally, option 4 will produce error "NameError: name 'pyhtml' is not defined".

5. Which of the following python codes will give error? [MCQ]

◯
```
import pyhtml
T = pyhtml.html(pyhtml.head(pyhtml.title("My website")),
       pyhtml.body(pyhtml.h1("This is the first heading"),
              pyhtml.h1("This is the second heading")))
out = T.render()
print(out)
```

◯
```
import pyhtml as h
T = h.html(h.head(h.title("My website")),
           h.body(h.h1("This is the first heading"),
                  h.h1("This is the second heading")))
out = T.render()
print(out)
```

◯
```
from pyhtml import *
T = html(head(title("My website")),
       body(h1("This is the first heading"),
              h1("This is the second heading")))
out = T.render()
print(out)
```

√
```
from pyhtml import *
T = pyhtml.html(pyhtml.head(pyhtml.title("My website")),
       pyhtml.body(pyhtml.h1("This is the first heading"),
              pyhtml.h1("This is the second heading")))
out = T.render()
print(out)
```

Solution: Options 1, 2 and 3 are the different ways which can be used to import and use pyhtml. In option 4, all the functions of pyhtml are already called. Therefore, there is no need to initiate a tag by writing pyhtml.<tag name>().

6. What will be the correct syntax for nesting a tag and a tag in a <body> tag using pyhtml library? [MCQ]

◯
```
import pyhtml as ph
file = ph.html(ph.body(ph.div("This is div.")),
       ph.body(ph.span("This is span.")))
```

√
```
import pyhtml as ph
file = ph.html(ph.body(ph.div("This is div."),
       ph.span("This is span.")))
```

◯
```
import pyhtml as ph
file = ph.html(ph.body(ph.span("This is div.")),
       ph.body(ph.div("This is span.")))
```

◯
```
import pyhtml as ph
file = ph.body(ph.div("This is div."),
       ph.span("This is span."))
```

Solution: The correct syntax for nesting a tag and tag in a <body> tag is given by:

```
import pyhtml as ph
file = ph.html(ph.body(ph.div("This is div."),
       ph.span("This is span.")))
```

7. What will be the output of the following Python code?

```
from jinja2 import Template
my_statement = Template("The special series is: {% for n in
                        range(1,15)%} {{n%3}}" "{% endfor %}")
out = my_statement.render()
print(out)
```

[MCQ]

◯ The special series is: 1 0 1 0 1 0 1 0 1 0 1 0 1 0

√ The special series is: 1 2 0 1 2 0 1 2 0 1 2 0 1 2

◯ The special series is: 0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0 5.5 6.0 6.5 7.0

◯ The special series is: 0.33 0.67 1.0 1.33 1.67 2.0 2.33 2.67 3.0 3.33 3.67 4.0 4.33 4.67

Solution: In the Template function, the {% %} is a block used to provide conditional statements and the content to be printed is given inside {{ }}. In Python, the expression n%3 will return the remainder when n is divided by 3.

8. A Python string template is given below:

```
from string import Template
my_statement = Template("All squares are $color1, all circles are $color2
                        and all triangles are $color3.")
```

Which of the following substitute statements will render the output string as "All squares are blue, all circles are red, and all triangles are green."? [MCQ]

- ◯ `out = my_statement.substitute(color1 = blue, color2 = red, color3 = green)`
- ◯ `out = my_statement.substitute(color1 = "blue", color2 = "green", color3 = "red")`
- √ `out = my_statement.substitute(color1 = "blue", color2 = "red", color3 = "green")`
- ◯ `out = my_statement.substitute(color1 = "red",color2 = "green", color3 = "yellow")`

Solution: The Template function of the string module allows the dummy content denoted by $variable in the output string to be replaced by the actual content. It takes the actual content through the substitute method, whose argument is a dictionary with key = dummy variable and value = actual content.

9. Which of the following Python codes will generate the HTML code given below? [MCQ]

```
<!DOCTYPE html>
<html>
    <head>
      <title>
         Modern Application Development
      </title>
    </head>
    <body>
        <h1>
            This is First level heading.
        </h1>
        <div>
          <h2>
             This is level 2 heading.
          </h2>
          <div>
             This is a nested div section.
```

```
        </div>
      </div>
      <div>
        <h3>
          This is some more text.
        </h3>
        <p>
          This is a paragraph.
        </p>
      </div>
    </body>
</html>
```

◯
```
import pyhtml as h
t=h.html(h.head(h.title("Modern Application Development")
),
h.body(
   h.h1("This is first level heading"),
   h.div(h.h2("This is level 2 heading"),
   h.div("This is a nested div section"),
   h.div(h.h3('This is some more text'),
   h.p("This is a paragraph")),
))
print(t.render())
```

√
```
import pyhtml as h
t=h.html(h.head(h.title("Modern Application Development")
),
h.body(
   h.h1("This is first level heading"),
   h.div(h.h2("This is level 2 heading"),
   h.div("This is a nested div section")),
   h.div(h.h3('This is some more text'),
   h.p("This is a paragraph")),
))
print(t.render())
```

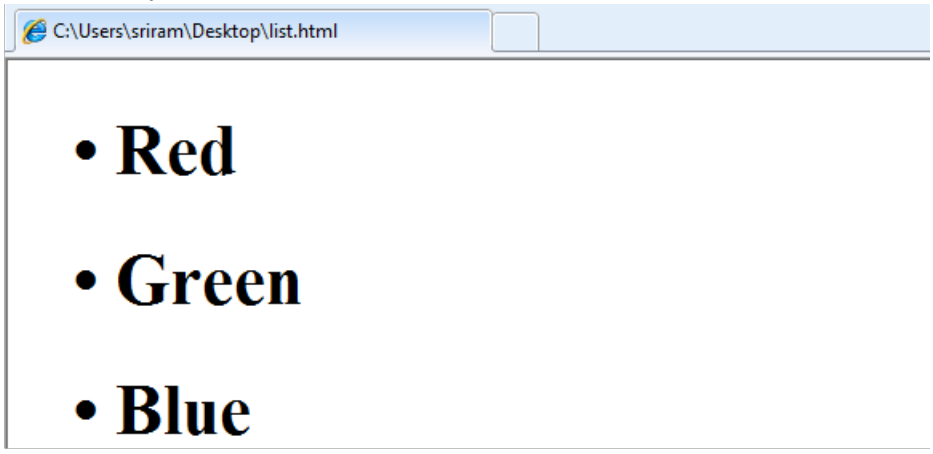
◯
```
import pyhtml as h
t=h.html(h.head(h.title("Modern Application Development")
)
h.body(
   h.h1("This is first level heading"),
   h.div(h.h2("This is level 2 heading"),
   h.div("This is a nested div section")),
   h.div(h.h3('This is some more text'),
   h.p("This is a paragraph")),
```

```
))
print(t.render())
```

◯
```
import pyhtml as h
t=h.html(h.head(h.title("Modern Application Development")
),
h.body(
   h.h1("This is first level heading"),
   h.div(h.h2("This is level 2 heading"),
   h.div("This is a nested div section")),
   h.div(h.h3('This is some more text'),
   h.p("This is a paragraph")),
)
print(t.render())
```

Solution:
For options (i), (iii) and (iv), in some places the commas are misplaced while in other places the brackets are missing. So, if we compile them, they would either not generate a valid HTML code or throw error. This does not happen in option (ii). It produces a correct HTML document as depicted in question. Hence, option II is correct.

10. Identify the correct pyhtml code that will generate the following HTML output when rendered by a browser. [MCQ]



◯
```
from pyhtml import *

t = html(
    body(
        ul(h1(li('Red'))),
        ol(h1(li('Green'))),
        ul(h1(li('Blue')))
```

```
        )
        )
print (t.render())
```

√
```
from pyhtml import *

t = html(
    body(
        ul(h1(li('Red')),h1(li('Green')),h1(li('Blue')))
        )
        )

print (t.render())
```

◯
```
from pyhtml import *

t = html(
    body(
        ul(h1(li('Red'))),
        ul(h1(li('Green'))),
        ul(h1(li('Blue')))
        )
        )
print (t.show())
```

◯ None of the above

Solution:
Option 1: The first code snippet will result in wrong output, as an ordered list is used.
Option 2: The second code snippet is the correct code and will result in the same view as in the given image.
Option 3: The third code snippet is not valid, as there is no such method called show().

11. Which of the following Python codes will generate the HTML code given below? [MCQ]

```
<!DOCTYPE html>
<html>
  <head>
    <title>
      Table
    </title>
  </head>
  <body>
    <dl>
      <dt>
        HTML
      </dt>
      <dd>
        Hyper Text Markup Language
      </dd>
      <dt>
        CSS
      </dt>
      <dd>
        Cascading Style Sheets
      </dd>
    </dl>
  </body>
</html>
```

√
```
from pyhtml import *

t = html(
    head(
        title('Table'),
        ),
    body(
        dl(dt('HTML'),dd('Hyper Text Markup Language')
           dt('CSS'),dd('Cascading Style Sheets')
          ),
        )
        )

print (t.render())
```

◯
```
from pyhtml import *
t = html(
    head(
```

```
        title('Definition List'),
     ),
    body(

           dl(dt('HTML'),dd('Hyper Text Markup Language')),
           dl(dt('CSS'),dd('Cascading Style Sheets'))

    )
)
print (t.render())
```

◯
```
from pyhtml import *

t = html(
    head(
        title('Table'),

    ),
    body(

           dd('HTML'),dt('Hyper Text Markup Language')
           ,dd('CSS'),dt('Cascading Style Sheets')

    )
)

print (t.render())
```

◯
```
from pyhtml import *

t = html(
    head(
        title('Table'),

    ),
    body(

           dl(dt('CSS'),dt('Cascading Style Sheets'),
           dt('HTML'),dt('Hyper Text Markup Language')),

    )
)
print (t.render())
```

Solution:
Option 1: Generates the required HTML code.
Option 2: Invalid because two dl() are written separately.
Option 3: Invalid because dt() and dd() order is incorrect.
Option 4: Invalid because no dd() is used.

12. Which of the following Python codes will generate the HTML document below? [MCQ]

```
<!DOCTYPE html>
<html>
  <head>
    <title>
    My html document
    </title>
  </head>
  <body>
    <div>
      <div>
        <div>
          <div>
            <h1>
               Hello World 1
            </h1>
          </div>
        </div>
        <p>
           My name is rohan verma
        </p>
        <h2>
           Hello World 2
        </h2>
      </div>
    </div>
  </body>
</html>
```

◯
```
from pyhtml import *
t=html(head(title('My html document')),
body(
  div(div(div(h1('Hello World 1')))),
  p('My name is rahul verma.'),
  h2('Hello World 2')))
)
```

```
)
print(t.render())
```

√
```
from pyhtml import *
t=html(head(title('My html document')),
body(
  div(div(div(div(h1('Hello World 1')))),
  p('My name is rahul verma.'),
  h2('Hello World 2')))
)
)
print(t.render())
```

◯
```
from pyhtml import *
t=html(head(title('My html document')),
body(
  div(div(div(div(h1('Hello World 1')))
  p('My name is rahul verma.')
  h2('Hello World 2')))
)
)
print(t.render())
```

◯ None of these

Solution: This question is based on the Python library called pyhtml to generate a valid HTML file using Python. (Reference: Timeframe 3:40 in Lecture 3.6, Tools-part II)