CSE 505 Spring 2017

Assignment 4: Higher-Order Functions and Advanced Control

Assigned: Fri, April 7, 2017
Due: Mon, April 17, 2017 (11:59 pm)
Note: This assignment may be done by a pair of students.

Problem 1. Consider the Python program discussed in Lecture 15:

```
def inorder_gen(tr, thk):
     def thk2(x):
          thk(x)
     if tr != None:
          inorder_gen(tr.left, thk2)
          thk(tr.value)
          inorder_gen(tr.right,thk2)

tree = node(10, node(20,None,None),node(30,node(40,None,None),None))

def thunk(x):
     print(x)

inorder_gen(tree, thunk)
```

Draw the stack diagram for this program at the point in execution when the run-time stack reaches its maximum height. You need to show the sequence of frames on the stack and their names, as well as their static and dynamic links. Save the diagram as a PNG file called **stack.png**.

Problem 2. Consider the following function definition in the C programming language for carrying out the summation expressed by the operator ∑ discussed in Lecture 15:

```
int sigma(int *k, int low, int high, int expr()) {
    int sum = 0;
    for (*k=low; *k<=high; (*k)++) {
        sum = sum + expr();
    }
    return sum;
}
```

Write a `main` program in the C programming language that makes repeated use of `sigma` in order to compute and print out the value of the following expression:

$$\sum_{i=0}^{4} \left(i * \sum_{j=0}^{4} \left((i+j) * \sum_{k=0}^{4} (j*k - i) \right) \right)$$

Place the code for `sigma` and `main` in a file **sigma.c**. Use the gcc compiler for testing your code.

Problem 3. Write a Python generator, called `flatten`, that takes as input a list of integers with arbitrary levels of nesting and *yields* one by one the integers present in the list. For example,

```
flatten([[[1],2],[[[[3]]]],[4,[5],[[6]]]])
```

should *yield* one by one the integers 1, 2, 3, 4, 5, and 6. Test the program by executing:

```
l = [[[1], 2],[[[[3]]]],[4,[5],[[6]]]
print [x for x in flatten(l)]
```

Create a file **flatten.py** containing the definition of `flatten` and the above two statements for testing `flatten`.

Problem 4. Translate your definition of flatten in problem 2 using higher-order functions following the method outlined in Lecture 15 for translating generators in terms of thunks. There should be one thunk function for each occurrence of the 'for' statement in your definition of flatten.

Note: the built-in list generator (in case you use it) should also be translated using a higher-order function.

Name the resulting program **flatten2.py**.

WHAT TO SUBMIT:

Prepare a top-level directory named *A4_UBITId1_UBITId2* if the assignment is done by two students; otherwise, name it as *A4_UBITId* if the assignment is done solo. (Order the *UBITId*'s in alphabetic order, in the former case.) In this directory, place **stack.png**, **sigma.c, flatten.py,** and **flatten2.y**.

Compress the directory and submit the resulting compressed file using the `submit_cse505` command. For more details regarding online submission, see

`Resources` → `Homeworks` → `Online_Submission_2017.pdf`.

End of Assignment #4