

CS206A Data Structure

HW2

Student ID: 20160253

name: 박예기 (Park ye gi)

Problem1(a).

ADT Stack is

objects : a finite ordered list with zero or more elements

functions :

for all $stack \in \text{Stack}$, $item \in \text{element}$, $max_stack_size \in \text{positive integer}$

Stack CreateS (max_stack_size) ::=

create an empty stack whose maximum size is max_stack_size

Boolean IsFull ($stack$, max_stack_size) ::=

if (number of elements in $stack == max_stack_size$)

return TRUE

else return FALSE

Boolean IsEmpty ($stack$) ::=

if ($stack == \text{CreateS}(max_stack_size)$)

return TRUE

else return FALSE

Stack Push ($stack$, $item$) ::=

if (IsFull($stack$)) **return** $stack_full$

else insert $item$ into top of $stack$ and **return** $stack$

Element Pop ($stack$) ::=

if (IsEmpty($stack$)) **return** $stack_empty$

else remove and **return** the $item$ on the top of the $stack$

Element Top ($stack$) ::=

if (IsEmpty($stack$)) then **return** $stack_empty$

else return the top element of $stack$

Problem1(b).

I used class to make object type Stack. I used class function to make function of Stack ADT.

To make CreateS() function, construct function(__init__) is used, and this function defines list for stack, top, and max_stack_size.

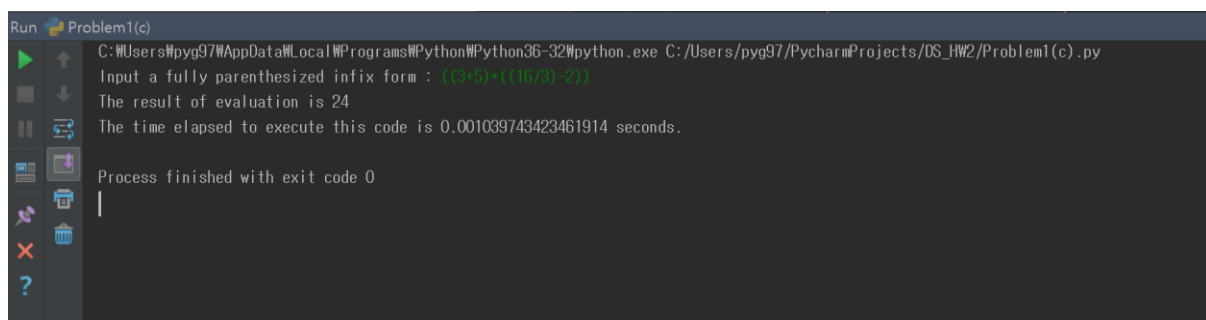
Function IsFull() and IsEmpty() determine the stack list is full or empty by comparing the top with max_stack_size-1 or zero. They give Boolean type output.

In function Push(), if the stack is full, it reports that stack is full, cannot add element. In other case, it makes item_list that has size 1 and has only the push item. Then it combines the list of stack and item_list. It gives the final stack list as output.

In function Pop(), if the stack is empty, it report that stack is empty, cannot pop element. In other case, it removes the element whose index is top by list slicing and returns the element.

In function Top(), if the stack is empty, it report that stack is empty, don't have top element. In other case, it returns the element whose index is top.

Problem1(c)



```
Run Problem1(c)
C:\Users\pyg97\AppData\Local\Programs\Python\Python36-32\python.exe C:/Users/pyg97/PycharmProjects/DS_HW2/Problem1(c).py
Input a fully parenthesized infix form : (12+5)*((10/3)-2)
The result of evaluation is 24
The time elapsed to execute this code is 0.001039743423461914 seconds.
Process finished with exit code 0
```

The input form is a fully parenthesized infix form and the output is an integer, the result of evaluation.

I used Stack class that I made in Problem1(b). I made two stacks, stack for Number and stack for Operation. I use for-loop to check every element of input. If the element is number, firstly check that the number is a single digit or not. Then If the element is "(", just pass. If the element is ")", pop two numbers from Number Stack and pop one operation from Operation Stack, then evaluate the operation with the two number. Then Push() the result of evaluation to Number Stack.

If the element is the last element, print the top element of Number Stack.

Problem 2(a)

ADT *PriorityQueue* is

objects : a finite ordered list with zero or more elements, which element has a priority.

functions :

for all $pq \in \text{PriorityQueue}$, $item \in \text{element}$, $max_pq_size \in \text{positive integer}$

PriorityQueue CreateQ (max_pq_size) ::=

create an empty priority queue whose maximum size is max_pq_size

Boolean IsFull (pq, max_pq_size) ::=

if (number of elements in $pq == max_pq_size$)

return TRUE

else return FALSE

Boolean IsEmpty (pq) ::=

if ($pq == \text{CreateQ}(max_pq_size)$)

return TRUE

else return FALSE

Element Top (pq) ::=

if (IsEmpty(pq)) then **return** pq_empty

else return the first element of highest priority elements in pq

Element Pop (pq) ::=

if (IsEmpty(pq)) **return** pq_empty

else remove and **return** the first element of the highest priority elements in pq

PriorityQueue Push ($pq, item$) ::=

if (IsFull(pq)) **return** pq_full

else insert $item$ into pq and **return** the resulting priority queue

Problem2(b)

I used class to make object type PriorityQueue. I used class function to make function of PriorityQueue ADT.

To make CreateQ() function, construct function(__init__) is used, and this function defines list for priority queue(pq), rear, and max_pq_size.

Function IsFull() and IsEmpty() determine the pq list is full or empty by comparing the rear with max_pq_size-1 or zero. They give Boolean type output.

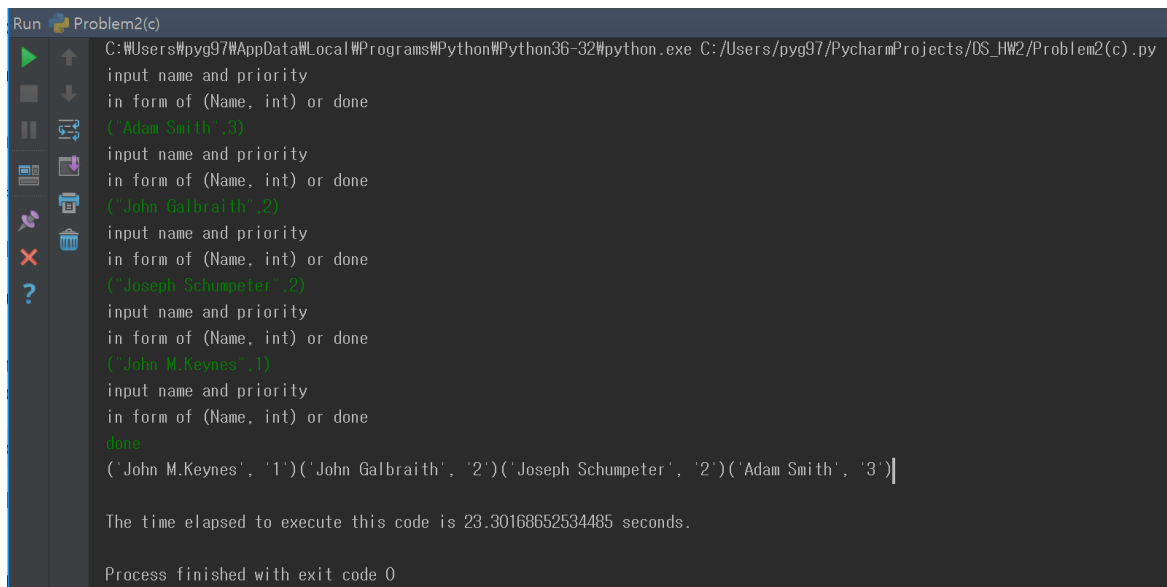
In function Push(), if the priority queue is full, it reports that priority queue is full, cannot add element. In other case, it makes item_list that has size 1 and has only the push item. Then by using

for-loop, compare the priority with the items in priority queue list. If the priority of item is less than that of element, (let whose index is 'i') , slicing the priority queue in accordance with "i"(two list : front_list and rear_list). Then combine the front_list, item_list, and rear_list. Finally, return the final priority queue list.

In function Pop(), if the priority queue is empty, it report that priority queue is empty, cannot pop element. In other case, it removes the element whose index is zero by list slicing and returns the element.

In function Top(), if the priority queue is empty, it report that priority queue is empty, don't have top element. In other case, it returns the element whose index is zero.

Problem2(c)



```
Run Problem2(c)
C:\Users\pyg97\AppData\Local\Programs\Python\Python36-32\python.exe C:/Users/pyg97/PycharmProjects/DS_HW2/Problem2(c).py
input name and priority
in form of (Name, int) or done
('Adam Smith', 3)
input name and priority
in form of (Name, int) or done
('John Galbraith', 2)
input name and priority
in form of (Name, int) or done
('Joseph Schumpeter', 2)
input name and priority
in form of (Name, int) or done
('John M Keynes', 1)
input name and priority
in form of (Name, int) or done
done
('John M Keynes', 1)('John Galbraith', 2)('Joseph Schumpeter', 2)('Adam Smith', 3)

The time elapsed to execute this code is 23.30168652534485 seconds.
Process finished with exit code 0
```

The input form is ("Name", priority) or done. The output is tuples.

I used Priority Queue class that I made in Problem2(b). I made object priority queue, whose name is reservation. By using while-loop, the code can get the input continuously. If the input is done, break the while-loop. If the input is ("Name", priority), then make a tuple (name, priority) by using list slicing. Then push the tuple into reservation priority queue. Finally, print out the lists.