Operating Systems (COE628)

Lab 1

Due the Week of January 25, 2021 (In Your lab Session), Late submission will not accepted

Objectives

- Review the C programming language.
- Learn how to use the Netbeans Integrated Development Environment (IDE) for C programs
- Implement a Priority Queue using a linked list.

Problem Description

You need to implement a Priority Queue using a linked list. A Priority Queue is used in Operating Systems kernel as part of the scheduling algorithm. Briefly, when a process cannot continue running (perhaps because it is waiting for some event such as input/output), the OS has to choose which waiting process that is ready to run gets control of the CPU. Part of this algorithm involves a priority queue where the ready processes are ordered by their priority. All things being equal, the highest priority task will be allowed to run. Template code is given to you. You need to implement the functions PQ_insert and PQ_delete functions.

You are given the header file **pri_queue.h:**

```
#ifndef PRI_QUEUE_H
#define PRI_QUEUE_H
typedef struct node {
    int priority;
    char * data;
    struct node * next;
} Node_t, * Node_ptr_t;
extern void PQ_insert( int, char *);
extern Node_ptr_t PQ_delete();
extern Node_ptr_t PQ_get_head();
extern int PQ_get_size();
#endif /* PRI_QUEUE_H */
```

This header file defines new data types:

- Node_t: A data structure for the nodes in the Priority Queue linked list. Each node contains its priority, data (a string) and a pointer to the next node in the list. (If a node is the last one on the list its "next" is NULL.)
- Node_ptr_t: A pointer to a Node_t structure.

The header file also contains the signatures for some functions:

- void PQ_insert (int, char *) : You have to implement this function. It inserts a new Node in the Priority Queue with the specified priority and data.
- Node_ptr_t PQ_delete(): You have to implement this function. It deletes and returns the highest priority Node.

You are also given the file **main. c:**

```
#include <stdio.h>
#include <stdlib.h>
#include <assert.h>
#include "pri_queue.h"

/* DO NOT MODIFY THIS FILE */

int main(int argc, char** argv) {

Node_ptr _t head;
assert(PQ_get _size()==0);
fprintf(stderr, "First assert worked!\n");
PQ_insert(0, "first node");
assert(n_get size() = 1);
fprintf(stderr, "Second assert worked!\n");
head = Pg_get head();
assert(head != NULL);
fprintf(stderr, "Third assert worked!\n");
assert(head->data == "first node");
fprintf(stderr, "Fourth assert worked!\n"); assert(head->priority = 0);
fprintf(stderr, "Fifth assert worked!\n"); PQ_insert(5, "abc");
head = PQ_get head();
assert(head->priority == 5);
fprintf(stderr, "Sixth assert worked!\n");
assert(head->next->priority == 0);
fprintf(stderr, "Seventh assert worked!\n"); PQ_insert(3, "def");
head = PQ_get head();
assert(head->priority = 5);
fprintf(stderr, "Eigth assert worked!\n");
assert(head->next->priority = 3);
fprintf(stderr, "Ninth assert worked!\n");
assert(head->next->next->priority = 0);
fprintf(stderr, "Tenth assert worked!\n");
PQinsert(7, "hij");
head = PQ_get head();
assert(head->priority = 7);
fprintf(stderr, "11th assert worked!\n");
assert(head->next->priority = 5);
fprintf(stderr, "12th assert worked!\n");
assert(head->next->next->priority = 3);
fprintf(stderr, "13th assert worked!\n");
assert(head->next->next->next->priority == 0);
fprintf(stderr, "14th assert worked! \n") ;
PQ_Lnsert(2, "pqr");
head = PQ_get head();
assert(head->priority = 7);
fprintf(stderr, "15th assert worked!\n");
assert(head->next->priority = 5);
fprintf(stderr, "16th assert workedl\n");
assert(head->next->next->priority = 3);
fprintf(stderr, "17th assert worked!\n");
assert(head->next->next->next->priority = 2);
fprintf(stderr, "18th assert worked!\n");
assert(head->next->next->next->next->priority = 0);
fprintf(stderr, "19th assert worked!\n");
return (EXIT_SUCCESS);
}
```

Finally, there is the file **pri_queue c:**

```
#include <stdlib.h>
#include <stdio.h>
#include "pri_queue.h"
/** @file pri_queue.c */
static Node_ptr_t head = NULL;
/**
* Insert a Node into a priority queue.
* @param priority
* @param data
* @author YOUR NAMX
*/
void PQ_Insert(int priority, char * data) {
//FIX THIS
}
/**
* Delete and return the node with the highest priority.
* @return The highest priority Node.
*/
Node_ptr_t PQ_delete() {
//FIX THIS
return NULL;
}
/**
* Do NOT modify this function.
* @return A pointer to the head of the list. (NULL if list is empty.)
*/
Node_ptr_t PQ_get _head (){
return head;
}
/**
* Do NOT modify this function.
* @return the number of items in the queue
*/
int PQ_get size(){
int size = 0;
for(Node_ptr_t tmp = head; tmp != NULL; tmp = tmp->next, size++)
return size;
}
```

Getting started

- Create a directory called COE628 with the command mkdir COE628
- Change to that directory with the command: cd COE628
- Download this zip file and save it in the COE628 directory.
- Unzip the labl.zip file with the command unzip labl.zip
- This creates a Netbeans project.
- Start Netbeans and open the project "labl" in your COE628 directory.
- Compile and run the project. It should compile and the first assertion should work.
- You now have to implement the insert and delete functions so that all of the assertions work.

Submit your lab

- Do you know how to create SSH session from home? If not , you can find instructions here :
  - Go to Remote Access:
  - https://www.ee.ryerson.ca/guides/user/
- Once you can remotely access lab computer.
  - Upload your submission folder (from local computer) to EE account. You will find screenshot for this in above mentioned link.
  - Go to the directory (through terminal of SSH) where your submission folder is placed on EE account.
  - Zip the submission folder:
    zip -r coe628_lab1.zip coe628_lab1
  - Submit the folder:
    submit coe628 lab1 coe628_lab1.zip