SUBJECT: NO: DATE: / /

```
#include <stdio.h>
#include <stdlib.h>
typedef struct node
{
void* dataPtr;
struct node* next;
} QUEUE_NODE;
typedef struct
{
QUEUE_NODE * front;
QUEUE_NODE * rear;
int count;
} QUEUE;
QUEUE * createQueue (void);
bool enqueue (QUEUE * queue, void * itemPtr);
void printQueue (QUEUE* stack);
int main (void)
{
QUEUE * queue1;
QUEUE * queue2;
QUEUE * queue3;
int *   numPtr;
int**   itemPtr;
queue1 = createQueue();
queue2 = createQueue();
queue3 = createQueue();
int i=1;
numPtr = (int*)malloc (sizeof(i));
*numPtr = i;
enqueue (queue1, numPtr);
```

SUBJECT : NO : DATE : / /

```
i = 3;
numPtr = (int*)malloc(sizeof(i));
*numPtr = i;
enqueue(queue1, numPtr);
i = 5;
numPtr = (int*)malloc(sizeof(i));
*numPtr = i;
enqueue(queue1, numPtr);
i = 7;
numPtr = (int*)malloc(sizeof(i));
*numPtr = i;
enqueue(queue2, numPtr);
i = 9;
numPtr = (int*)malloc(sizeof(i));
*numPtr = i;
enqueue(queue2, numPtr);
i = 11;
numPtr = (int*)malloc(sizeof(i));
*numPtr = i;
enqueue(queue2, numPtr);
i = 13;
numPtr = (int*)malloc(sizeof(i));
*numPtr = i;
enqueue(queue3, numPtr);
i = 15;
numPtr = (int*)malloc(sizeof(i));
*numPtr = i;
enqueue(queue3, numPtr);
```

SUBJECT: NO: DATE: / /

```
i=17;
numPtr = (int *) malloc(sizeof(i));
*numPtr = i;
enqueue(queue3, numPtr);
printf("Queue 1:\n");
printQueue(queue1);
printf("Queue2:\n");
printQueue(queue2);
printf("Queue3:\n");
printQueue(queue3);
return 0;
}
QUEUE * createQueue(void)
{
QUEUE * queue;
queue = (QUEUE *) malloc(sizeof(QUEUE));
if (queue)
{
queue -> front = NULL;
queue -> rear = NULL;
queue -> count = 0;
}
return queue;
}
bool enqueue(QUEUE * queue, void * itemPtr)
{
QUEUE_NODE* newPtr = (QUEUE_NODE*)malloc(sizeof(QUEUE_NODE));
newPtr -> dataPtr = itemPtr;
newPtr -> next = NULL;
```

SUBJECT : NO : DATE : / /

```
if (queue -> count == 0)
queue -> front = newPtr;
else
queue -> rear -> next = newPtr;
(queue -> count)++;
queue -> rear = newPtr;
return true;
}
QUEUE* destroyQueue (QUEUE *queue)
{
QUEUE_NODE* deletePtr;
if (queue)
{
while (queue -> front != NULL)
{
free (queue -> front -> dataPtr);
deletePtr = queue -> front;
queue -> front = queue -> front -> next;
free (deletePtr);
}
free (queue);
}
return NULL;
}
void printQueue (QUEUE *queue)
{
QUEUE_NODE* node = queue -> front;
printf ("Front => ");
while (node)
```

SUBJECT : NO : DATE : / /

```
{
printf ("%3d", *(int*)node->dataPtr);
node = node->next;
}
printf (" <= Rear\n");
return;
}
```