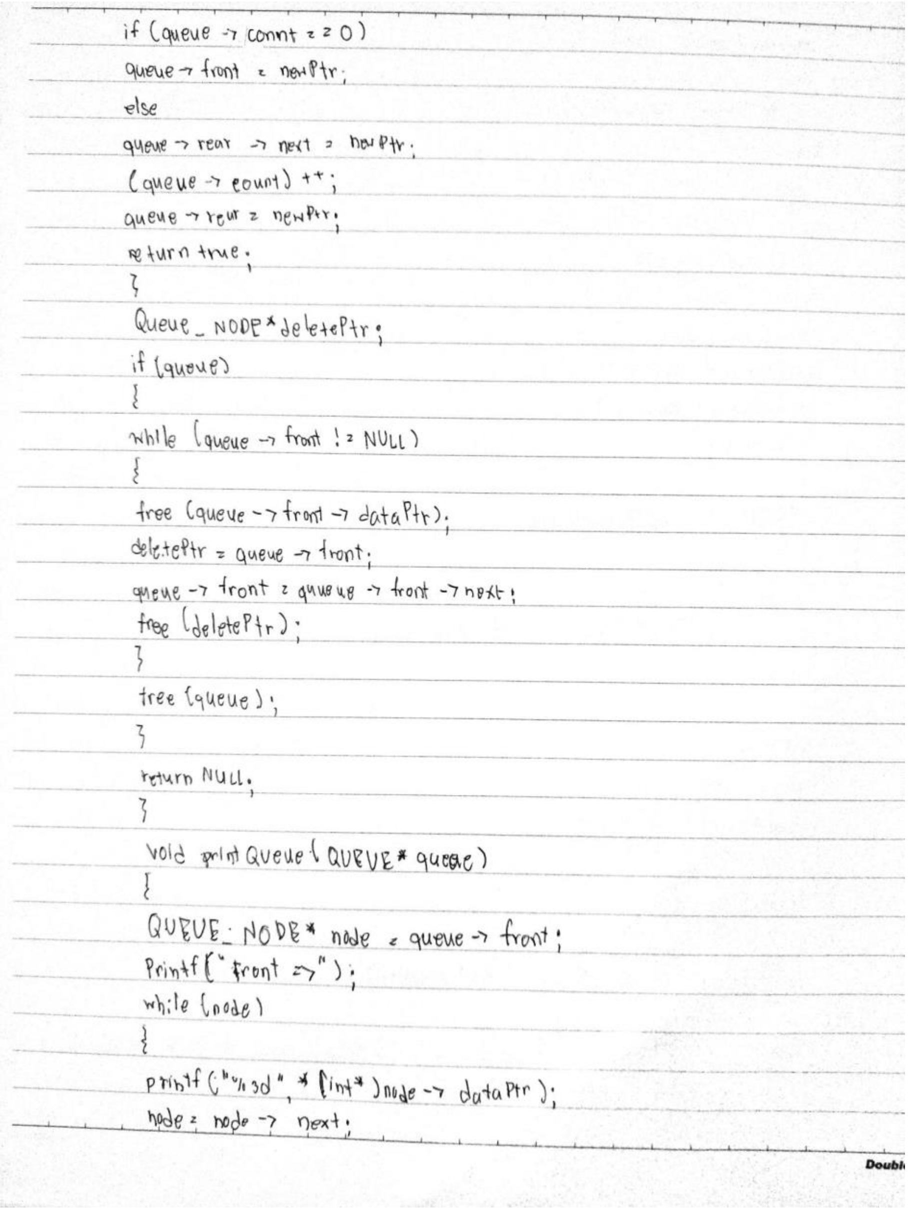
```
# include < stdio. h>
# include < stdio h>
  typedef struct node
  {
   void* dataptr;
   struct node * next;
  } Queue _ Node;
   typedef struct
  }
   QUEUE_NODE* front;
   QUEUE_NODE* rear;
   Int count
  } QUEUE
  QUEUE * create Queue (void);
  bool enqueue ( QUEUE * queue, void * itemPty) ;
  Void print Queue ( QUEUE* stack );
  int main (Void)
  { .
  // Local Definitions
  QUEUE * queue 1;
  QUEUE* queue 2;
  QUEUE* queue 3;
   int * numPtr;
   int ** itemPtr;
   queue1 = createQueue ();
   queue2 = CreateQueue ();
   queue3 = CreateQueue ();
   int i = 9;
```

```
numPtr = (int *) malloc(sizeof(i));
* numPtr =1;
enqueue(queue1, numptr);
i = 8;
numPtr = (int *) malloc(sizeof(i));
* numPtr = i;
enqueue(queue1, numPtr);
i = 7;
numPtr = (int *) malloc(sizeof(i));
* numPtr = i;
enqueue(queue1, numPtr);
i = 6;
numPtr = (int *) malloc(sizeof(i));
* numPtr = i;
enqueue(queue2, numPtr);
i = 5;
numPtr = (int *) malloc(sizeof(i));
* numPtr = i;
enqueue(queue2, numPtr);
i = 4;
numPtr = (int *) malloc(sizeof(i));
* numPtr = i;
enqueue(queue2, numptr);
i = 3;
numPtr = (int *) malloc(sizeof(i));
* numPtr = i;
enqueue(queue3, numPtr);
i = 2;
numPtr = (int *) malloc(sizeof(i));
* numPrt = i;
```



```
if (queue -> connt == 0)
queue -> front = newPtr;
else
queue -> rear -> next = newPtr;
(queue -> eount) ++;
queue -> reur = newPtr;
return true;
}
Queue_NODE*deletePtr;
if (queue)
{
while (queue -> front != NULL)
{
free (queue -> front -> dataPtr);
deletePtr = queue -> front;
queue -> front = queue -> front -> next;
free (deletePtr);
}
tree (queue);
}
return NULL;
}
void printQueue (QUEUE* queue)
{
QUEUE_NODE* node = queue -> front;
Printf(" front => ");
while (node)
{
printf("%3d", *(int*)node -> dataPtr);
node = node -> next;
```

```
enqueue (queue3, numPtr);
i = 1;
numPtr = (int *) malloc (sizeof (i));
* numPrt = i;
 enqueue ( queue 3, numPtr );
printf (" Queue 1:\n");
printQueue (queue 1);
printf (" Queue 2:\n");
 print Queue ( queue 2 );
printQueue (queue 3 :\n");
print Queue (queue 3 );
return 0;
}
 QUEUE* create Queue ( 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;
```

```
     }
      printf ("<= Rear\n");
       return;
      }
```