```c
#include <stdio.h>
#include <stdlib.h>

typedef struct queueNode {   /* self-referential structure */
    char data;
    struct queueNode* nextPtr;
} QueueNode_t;

/* function prototypes */
void printQueue(QueueNode_t*);
int isEmpty(QueueNode_t*);
char dequeue(QueueNode_t**, QueueNode_t**);
void enqueue(QueueNode_t**, QueueNode_t**, char);
void instructions(void);

int main()
{
    QueueNode_t  *headPtr = NULL, *tailPtr = NULL;
    int choice;
    char item;

    instructions();
    printf("? ");
    scanf("%d", &choice);

    while (choice != 3) {

        switch (choice) {

        case 1:
            printf("Enter a character: ");
            scanf("\n%c", &item);
            enqueue(&headPtr, &tailPtr, item);
            printQueue(headPtr);
            break;
        case 2:
            if (!isEmpty(headPtr)) {
                item = dequeue(&headPtr, &tailPtr);
                printf("%c has been dequeued.\n", item);
            }

            printQueue(headPtr);
            break;

        default:
            printf("Invalid choice.\n\n");
            instructions();
            break;
        }

        printf("? ");
        scanf("%d", &choice);
    }
```

```
54
55      printf("End of run.\n");
56      return 0;
57  }
58
59  void instructions(void)
60  {
61      printf("Enter your choice:\n"
62          "   1 to add an item to the queue\n"
63          "   2 to remove an item from the queue\n"
64          "   3 to end\n");
65  }
66
67  void enqueue(QueueNode_t** headPtr, QueueNode_t** tailPtr, char value)
68  {
69      QueueNode_t* newPtr;
70
71      newPtr = malloc(sizeof(QueueNode_t));
72
73      if (newPtr != NULL) {
74          newPtr->data = value;
75          newPtr->nextPtr = NULL;
76
77          if (isEmpty(*headPtr))
78              *headPtr = newPtr;
79          else
80              (*tailPtr)->nextPtr = newPtr;
81
82          *tailPtr = newPtr;
83      }
84      else
85          printf("%c not inserted. No memory available.\n",
86              value);
87  }
88
89  char dequeue(QueueNode_t** headPtr, QueueNode_t** tailPtr)
90  {
91      char value;
92      QueueNode_t* tempPtr;
93
94      value = (*headPtr)->data;
95      tempPtr = *headPtr;
96      *headPtr = (*headPtr)->nextPtr;
97
98      if (*headPtr == NULL)
99          *tailPtr = NULL;
100
101     free(tempPtr);
102     return value;
103 }
104
105 int isEmpty(QueueNode_t* headPtr)
106 {
```

```
107     return headPtr == NULL;
108 }
109
110 void printQueue(QueueNode_t* currentPtr)
111 {
112     if (currentPtr == NULL)
113         printf("Queue is empty.\n\n");
114     else {
115         printf("The queue is:\n");
116
117         while (currentPtr != NULL) {
118             printf("%c --> ", currentPtr->data);
119             currentPtr = currentPtr->nextPtr;
120         }
121
122         printf("NULL\n\n");
123     }
124 }
```