```c
/*****************************************************************************
 File name:  main.c
 Author:     Computer Science, Pacific University
 Date:       8/30/17
 Class:      CS300
 Assignment: Exam Stats
 Purpose:    To create a program that will give you the mean for a given set
 of exams.

 *****************************************************************************/

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

#define MAX_EXAM_SCORES 50
#define MIN_EXAM_SCORES 1
#define MAX_SCORE 100

static void getExamScores (int aExamScores[], int numExamScores);
static void getExamScore (int *pExamScore, int whichScore);
static void printExamScores (const int aExamScores[], int numExamScores);
static float calculateMean (const int aExamScores[], int numExamScores);

/*****************************************************************************
 Function:    main

 Description: Driver for entering exam scores and finding/printing the exam
 scores and the mean of the exam scores.

 Parameters:  None

 Returned:    Exit Status

 *****************************************************************************/

int main ()
{
  int numExamScores;
  int *pExamScores;

  printf ("Exam Stats \n\n");

  do
```

```
42  {
43    printf ("Enter Number of Exams: ");
44    scanf ("%d", &numExamScores);
45  } while (numExamScores < MIN_EXAM_SCORES || numExamScores >
  MAX_EXAM_SCORES);
46
47  pExamScores = (int *) malloc (sizeof(int) * numExamScores);
48
49  if (NULL == pExamScores)
50  {
51    printf ("Error: Cannot Allocate Memory\n\n");
52    exit (EXIT_FAILURE);
53  }
54
55  printf ("\n");
56  getExamScores (pExamScores, numExamScores);
57
58  printf ("\n");
59  printExamScores (pExamScores, numExamScores);
60
61  printf ("\n\n");
62  printf ("Mean is %6.2f", calculateMean (pExamScores, numExamScores));
63
64  free (pExamScores);
65
66  return EXIT_SUCCESS;
67}
68
69/*****************************************************************************
70 Function:    getExamScores
71
72 Description: Enters exam scores into an array and then calls insert which
73 places the exams in increasing order;
74
75 Parameters:  aExamScores    - an array of exam scores
76 numExamScores  - the number of exams scored
77
78 Returned:    None
79
  *****************************************************************************/
80
81static void getExamScores (int aExamScores[], int numExamScores)
82{
```

```c
  int i;

  for (i = 0; i < numExamScores; ++i)
  {
    getExamScore (&aExamScores[i], i + 1);
  }
}

/*****************************************************************************
 Function:    getExamScore

 Description: Gets an exam score and checks it's between the allowed values.

 Parameters:  pExamScore - a pointer to an exam score
 whichScore - the number of the scored exam

 Returned:    None

 *****************************************************************************/

static void getExamScore (int *pExamScore, int whichScore)
{
  do
  {
    printf ("Enter Exam Score %d: ", whichScore);
    scanf ("%d", pExamScore);
  } while (*pExamScore < 0 || *pExamScore > MAX_SCORE);
}

/*****************************************************************************
 Function:    printExamScores

 Description: Displays the exam scores in order from the array

 Parameters:  aExamScores    - an array of exam scores
 numExamScores  - the number of exams scored

 Returned:    None

 *****************************************************************************/

static void printExamScores (const int aExamScores[], int numExamScores)
{
```

```
124   const int NUMS_PER_ROW = 5;
125   int i;
126
127   printf ("Exam Summary Output\n\n");
128
129   printf ("Exam Scores\n");
130
131   for (i = 0; i < numExamScores; ++i)
132   {
133     if (0 == i % NUMS_PER_ROW)
134     {
135       printf ("\n");
136     }
137
138     printf ("%4d", aExamScores[i]);
139   }
140 }
141
142 /*****************************************************************************
143  Function:    calculateMean
144
145  Description: Calculates the mean of an array of exam scores
146
147  Parameters:  aExamScores    - an array of exam scores
148  numExamScores  - the number of exams scored
149
150  Returned:    the average of the exam scores;
151
    *****************************************************************************/
152
153 static float calculateMean (const int aExamScores[], int numExamScores)
154 {
155   int i;
156   float mean = 0.0;
157   float sumOfScores = 0;
158
159   for (i = 0; i < numExamScores; ++i)
160   {
161     sumOfScores += aExamScores[i];
162   }
163
164   if (numExamScores != 0)
165   {
```

```
166     mean = sumOfScores / numExamScores;
167   }
168
169   return mean;
170 }
171
```