```c
#include <stdio.h>  // For printf
#include <stdlib.h> // For rand() and srand()
#include <time.h>   // For time() (used to seed the random number
generator)

void generateDate(float arr[], int n, int min, int max) {
  for (int i = 1; i < n; i++) {
    arr[i] = min + ((float)rand() / RAND_MAX) * (max - min);
  }
}

int findMin(float arr[], int n) {
  float min = arr[1];
  int maxIndex = 1;
  for (int i = 1; i < n; i++) {
    if (arr[i] > min) {
      min = arr[i];
      maxIndex = i;
    }
  }
  return maxIndex;
}

int linearSearch(float arr[], int n, float target) {
  for (int i = 0; i < n; i++) {
    if (arr[i] >= target) {
      return i;
    }
  }
  return -1;
}

void sort(float arr[], int len) {
  for (int i = 0; i < len; i++) {
    float largest = arr[i];
    int largestIndex = i;
    for (int j = i; j < len; j++) {
      if (arr[j] < largest) {
        largest = arr[j];
        largestIndex = j;
      }
    }
    arr[largestIndex] = arr[i];
    arr[i] = largest;
  }
}

int findWithBinary(float arr[], int len, float target) {
  sort(arr, len);

  int low = 0, high = len - 1;

  while (low <= high) {

    int mid = (low + high) / 2;
    if (arr[mid] >= target) {
      if (mid == 0) {
        return mid;
      }
      if (arr[mid - 1] >= target) {
        high = mid - 1;
      } else {
        return mid;
      }
    } else {
      low = mid + 1;
    }
  }

  return -1;
}

int findMax(float arr[], int n) {
  float max = arr[1];
  int minIndex = 1;
  for (int i = 1; i < n; i++) {
    if (arr[i] < max) {
      max = arr[i];
      minIndex = i;
      printf("%f \n", arr[i]);
    }
  }
  printf("%f \n", arr[1]);

  return minIndex;
}
int main(int argc, char const *argv[]) {
  int n = 10001;
  float arr[n];
  float pressureArr[n];

  generateDate(arr, n, 21, 50);
  generateDate(pressureArr, n, 951, 1050);
  double duration;
  clock_t start, end; // typedef of a numeric type: represent running
time

  // start = clock();  //returns processor clock time since the program
is
  // started

  // int minIndx=findMin(arr,n);

  // end=clock();
  start = clock(); // returns processor clock time since the program is
started
  int maxIndx = findMax(pressureArr, n);
  printf("max is %d \n", maxIndx);
  end = clock();
  duration = (((double)(end - start)) /
              CLOCKS_PER_SEC); // no. of clock ticks per second

  printf("For the input size=%d, Time required to find minimum value in a
"
         "list=%lf seconds\n",
         n, duration);
  return 1;
}
```