BINARY HEAP AND HEAP SORT PROBLEMS

1) 14, 20, 33, 15, 11

Min-Heap Implementation:

```
#include <stdio.h>

#define MAX_SIZE 100

void swap(int *a, int *b) {
    int temp = *a;
    *a = *b;
    *b = temp;
}

void heapifyUpMin(int heap[], int index) {
    int parent = (index - 1) / 2;
    if (index > 0 && heap[index] < heap[parent]) {
        swap(&heap[index], &heap[parent]);
        heapifyUpMin(heap, parent);
    }
}

void insertMinHeap(int heap[], int *size, int value) {
    if (*size == MAX_SIZE) {
        printf("Heap is full!\n");
        return;
    }
    heap[*size] = value;
    (*size)++;
    heapifyUpMin(heap, *size - 1);
}

void printHeap(int heap[], int size) {
    for (int i = 0; i < size; i++) {
        printf("%d ", heap[i]);
    }
    printf("\n");
}

int main() {
    int heap[MAX_SIZE];
    int size = 0;
```

```
    int numbers[] = {14, 20, 33, 15, 1};
    int n = sizeof(numbers) / sizeof(numbers[0]);

    for (int i = 0; i < n; i++) {
        insertMinHeap(heap, &size, numbers[i]);
    }

    printf("Min-Heap: ");
    printHeap(heap, size);

    return 0;
}
```

Max-Heap Implementation:

```
#include <stdio.h>

#define MAX_SIZE 100

void swap(int *a, int *b) {
    int temp = *a;
    *a = *b;
    *b = temp;
}

void heapifyUpMax(int heap[], int index) {
    int parent = (index - 1) / 2;
    if (index > 0 && heap[index] > heap[parent]) {
        swap(&heap[index], &heap[parent]);
        heapifyUpMax(heap, parent);
    }
}

void insertMaxHeap(int heap[], int *size, int value) {
    if (*size == MAX_SIZE) {
        printf("Heap is full!\n");
        return;
    }
    heap[*size] = value;
    (*size)++;
    heapifyUpMax(heap, *size - 1);
}
```

```
void printHeap(int heap[], int size) {
    for (int i = 0; i < size; i++) {
        printf("%d ", heap[i]);
    }
    printf("\n");
}

int main() {
    int heap[MAX_SIZE];
    int size = 0;

    int numbers[] = {14, 20, 33, 15, 1};
    int n = sizeof(numbers) / sizeof(numbers[0]);

    for (int i = 0; i < n; i++) {
        insertMaxHeap(heap, &size, numbers[i]);
    }

    printf("Max-Heap: ");
    printHeap(heap, size);

    return 0;
}
```

HEAP SORT:

Min-Heap Sort Implementation:

```
#include <stdio.h>

void swap(int *a, int *b) {
    int temp = *a;
    *a = *b;
    *b = temp;
}

void heapifyDownMin(int heap[], int size, int index) {
    int smallest = index;
    int left = 2 * index + 1;
    int right = 2 * index + 2;

    if (left < size && heap[left] < heap[smallest]) {
```

```
        smallest = left;
    }

    if (right < size && heap[right] < heap[smallest]) {
        smallest = right;
    }

    if (smallest != index) {
        swap(&heap[index], &heap[smallest]);
        heapifyDownMin(heap, size, smallest);
    }
}

void buildMinHeap(int heap[], int size) {
    for (int i = size / 2 - 1; i >= 0; i--) {
        heapifyDownMin(heap, size, i);
    }
}

void heapSortMin(int heap[], int size) {
    buildMinHeap(heap, size);
    for (int i = size - 1; i >= 0; i--) {
        swap(&heap[0], &heap[i]);
        heapifyDownMin(heap, i, 0);
    }
}

void printArray(int arr[], int size) {
    for (int i = 0; i < size; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
}

int main() {
    int numbers[] = {14, 20, 33, 15, 1};
    int size = sizeof(numbers) / sizeof(numbers[0]);

    heapSortMin(numbers, size);

    printf("Sorted array using Min-Heap: ");
    printArray(numbers, size);

    return 0;
```

```
}
```

Max-Heap Sort Implementation:

```
#include <stdio.h>

void swap(int *a, int *b) {
    int temp = *a;
    *a = *b;
    *b = temp;
}

void heapifyDownMax(int heap[], int size, int index) {
    int largest = index;
    int left = 2 * index + 1;
    int right = 2 * index + 2;

    if (left < size && heap[left] > heap[largest]) {
        largest = left;
    }

    if (right < size && heap[right] > heap[largest]) {
        largest = right;
    }

    if (largest != index) {
        swap(&heap[index], &heap[largest]);
        heapifyDownMax(heap, size, largest);
    }
}

void buildMaxHeap(int heap[], int size) {
    for (int i = size / 2 - 1; i >= 0; i--) {
        heapifyDownMax(heap, size, i);
    }
}

void heapSortMax(int heap[], int size) {
    buildMaxHeap(heap, size);
    for (int i = size - 1; i >= 0; i--) {
        swap(&heap[0], &heap[i]);
        heapifyDownMax(heap, i, 0);
    }
```

```
}

void printArray(int arr[], int size) {
    for (int i = 0; i < size; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
}

int main() {
    int numbers[] = {14, 20, 33, 15, 1};
    int size = sizeof(numbers) / sizeof(numbers[0]);

    heapSortMax(numbers, size);

    printf("Sorted array using Max-Heap: ");
    printArray(numbers, size);

    return 0;
}
```