```
#include <iostream>

int getMax(int arr[], int size) {
    int maxVal = arr[0];
    for (int i = 1; i < size; i++) {
        if (arr[i] > maxVal) {
            maxVal = arr[i];
        }
    }
    return maxVal;
}
```

此函数用于获取数组中的最大值。它遍历数组并比较每个元素与当前的最大值，然后返回最大值。

```
void countingSort(int arr[], int size, int exp) {
    int output[size];    // 存储排序结果的临时数组
    int count[10] = {0}; // 存储计数的数组

    // 统计元素出现次数
    for (int i = 0; i < size; i++) {
        count[(arr[i] / exp) % 10]++;
    }

    // 计算累加次数
    for (int i = 1; i < 10; i++) {
        count[i] += count[i - 1];
    }

    // 构建排序结果
    for (int i = size - 1; i >= 0; i--) {
        output[count[(arr[i] / exp) % 10] - 1] = arr[i];
        count[(arr[i] / exp) % 10]--;
    }

    // 将排序结果复制到原数组
    for (int i = 0; i < size; i++) {
        arr[i] = output[i];
    }
}
```

这是计数排序函数，它接收数组、数组大小和当前的位数作为参数。它使用计数数组和临时数组来执行计数排序算法。首先，它统计每个数字出现的次数，将其存储在计数数组中。然后，它计算累积次数，用于确定每个数字在排序结果中的位置。接下来，它根据当前位上的数字，将元素放入临时数组的正确位置。最后，将排序结果复制回原数组。

```
void radixSort(int arr[], int size) {
    int maxVal = getMax(arr, size);
```

```
    // 对每位进行计数排序
    for (int exp = 1; maxVal / exp > 0; exp *= 10) {
        countingSort(arr, size, exp);
    }
}
```

这是基数排序函数，它接收数组和数组大小作为参数。它首先通过调用getMax函数来获取数组中的最大值。然后，它根据最大值的位数，从个位开始，依次对每个位进行计数排序。通过多次的计数排序操作，最终得到有序的数组。

```
int main() {
    int arr[] = {64, 34, 25, 12, 22, 11, 90};
    int size = sizeof(arr) / sizeof(arr[0]);

    radixSort(arr, size);

    std::cout << "排序结果: ";
    for (int i = 0; i < size; i++) {
        std::cout << arr[i] << " ";
    }
    std::cout << std::endl;

    return 0;
}
```

这是主函数，用于测试基数排序算法。它创建一个包含待排序元素的数组，并计算数组的大小。然后，它调用radixSort函数对数组进行排序。最后，它输出排序结果。

希望这些注释和解释能帮助您理解基数排序算法的思路和实现。