Design and Analysis of Algorithms

Week-6

CH.SC.U4CSE24121

KURRA VENKATA GOKUL

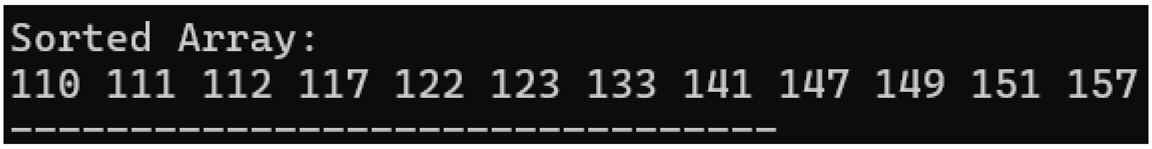
Quick sort:

(starting number as pivot element)

Code:

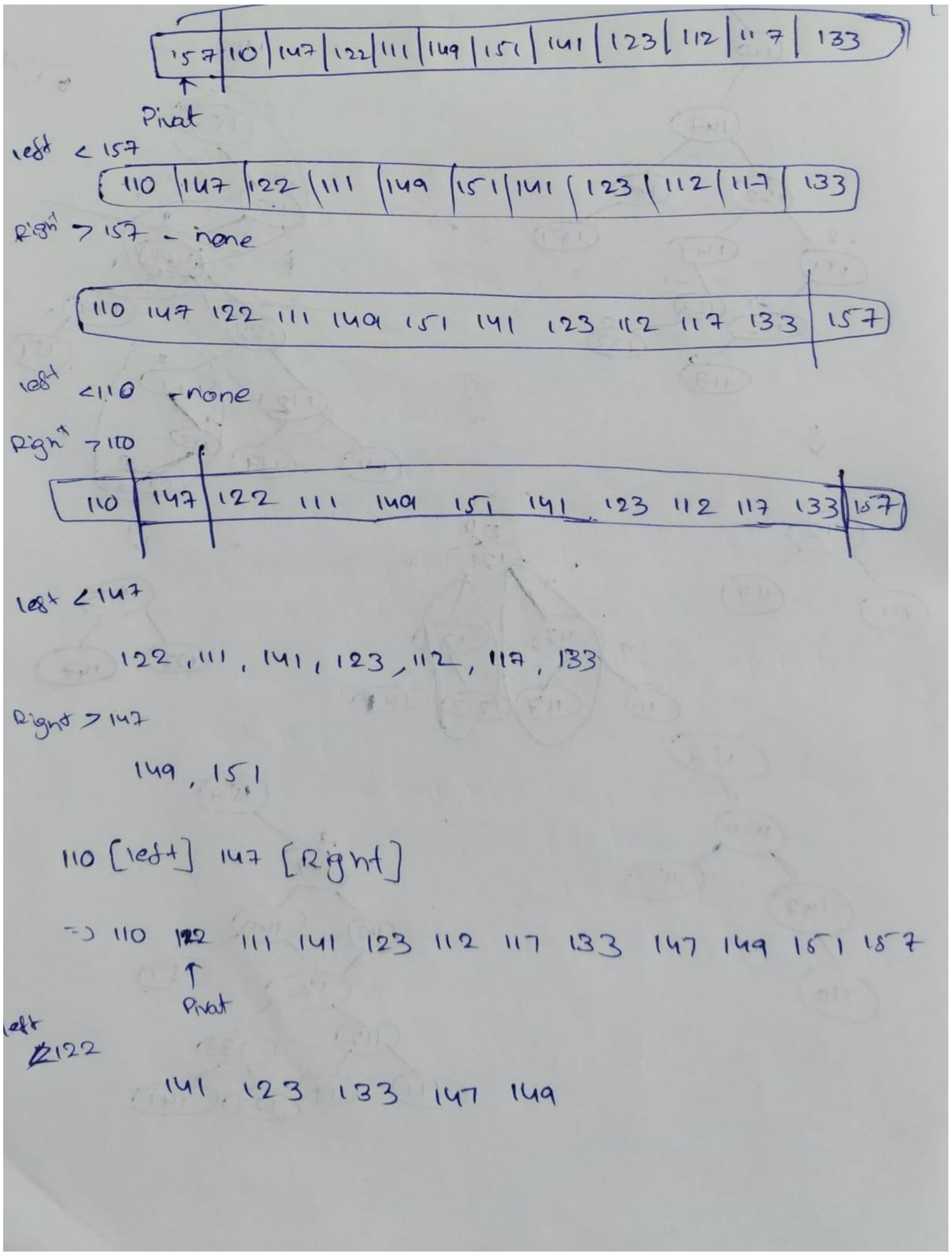
```
//CH.SC.U4CSE24121
#include <stdio.h>
int partitionFirst(int a[], int low, int high)
{
    int pivot, i, j, temp;
    pivot = a[low];
    i = low + 1;
    j = high;

    while (i <= j)
    {
        while (a[i] <= pivot && i <= high)
            i++;
        while (a[j] > pivot)
            j--;

        if (i < j)
        {
            temp = a[i];
            a[i] = a[j];
            a[j] = temp;
        }
    }
    temp = a[low];
    a[low] = a[j];
    a[j] = temp;

    return j;
}
void quickSortFirst(int a[], int low, int high)
{
    int p;
    if (low < high)
    {
        p = partitionFirst(a, low, high);
        quickSortFirst(a, low, p - 1);
        quickSortFirst(a, p + 1, high);
    }
}
int main()
{
    int a[12] = {157,110,147,122,111,149,151,141,123,112,117,133};
    int i;
    quickSortFirst(a, 0, 11);
    printf("Sorted Array:\n");
    for (i = 0; i < 12; i++)
        printf("%d ", a[i]);
    return 0;
}
```

Output:



```
Sorted Array:
110 111 112 117 122 123 133 141 147 149 151 157
--------------------------------
```

Handwritten:



157 | 110 | 147 | 122 | 111 | 149 | 151 | 141 | 123 | 112 | 117 | 133

↑
Pivot

left < 157

110 | 147 | 122 | 111 | 149 | 151 | 141 | 123 | 112 | 117 | 133

Right > 157 - none

110 147 122 111 149 151 141 123 112 117 133 | 157

left <110 - none

Right >110

110 | 147 | 122 111 149 151 141 123 112 117 133 | 157

left <147

122 , 111 , 141 , 123 , 112 , 117 , 133

Right >147

149 , 151

110 [left] 147 [Right]

=> 110 122 111 141 123 112 117 133 147 149 151 157

↑
Pivot

left
<122

141 , 123 133 147 149

110 (left] 147 (Right] 157

left < 122

111 112 117

Right > 122

~~123~~ 141 123 133 149

110 [left] 122 (Right]

110 111 112 117 122 | 141 123 133 147 149 151 157

↓ Pivot

< 141

123 133

> 141

147 149 151 157

110 111 112 117 122 123 133 141 | 147 149 151 157

↓ Pivot

< 147 - none

> 147

149 151 157

110 111 112 117 122 123 133 141 147 | 149 151 157

↓ Pivot

left Pivot Righ

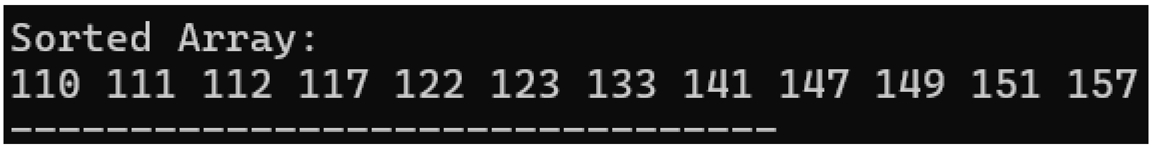
Quick sort:

(last number as pivot element)

Code:

```
//CH.SC.U4CSE24121

#include <stdio.h>

int partitionLast(int a[], int low, int high)

{

    int pivot, i, j, temp;

    pivot = a[high];

    i = low - 1;

    for (j = low; j < high; j++)

    {

        if (a[j] <= pivot)

        {

            i++;

            temp = a[i];

            a[i] = a[j];

            a[j] = temp;

        }

    }

    temp = a[i + 1];

    a[i + 1] = a[high];

    a[high] = temp;

    return i + 1;

}

void quickSortLast(int a[], int low, int high)

{

    int p;
    if (low < high)
    {
        p = partitionLast(a, low, high);
        quickSortLast(a, low, p - 1);
        quickSortLast(a, p + 1, high);
    }
}
int main()
{
    int a[12] = {157,110,147,122,111,149,151,141,123,112,117,133};
    int i;
    quickSortLast(a, 0, 11);
    printf("Sorted Array:\n");
    for (i = 0; i < 12; i++)
        printf("%d ", a[i]);
    return 0;
}
```

Output:



```
Sorted Array:
110 111 112 117 122 123 133 141 147 149 151 157
--------------------------------
```

Handwritten:

Last Element as PIVOT

157 110 147 122 111 149 151 141 123 112 117 133

↑
Pivot

122 < 133

110 157 147 122 111 149 151 141 123 112 117 133

110 122 147 157 111 149 151 141 123 112 117 133

111 < 133

110 122 111 157 147 149 151 141 123 112 117 133

123 < 133

110 122 111 123 147 149 151 141 157 112 117 133

112 < 133

117 < 133

110 122 111 123 112 117 151 141 157 147 149 133

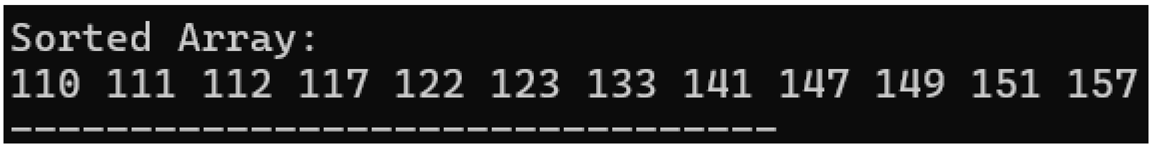
Quick sort:

(Middle number as pivot element)

Code:

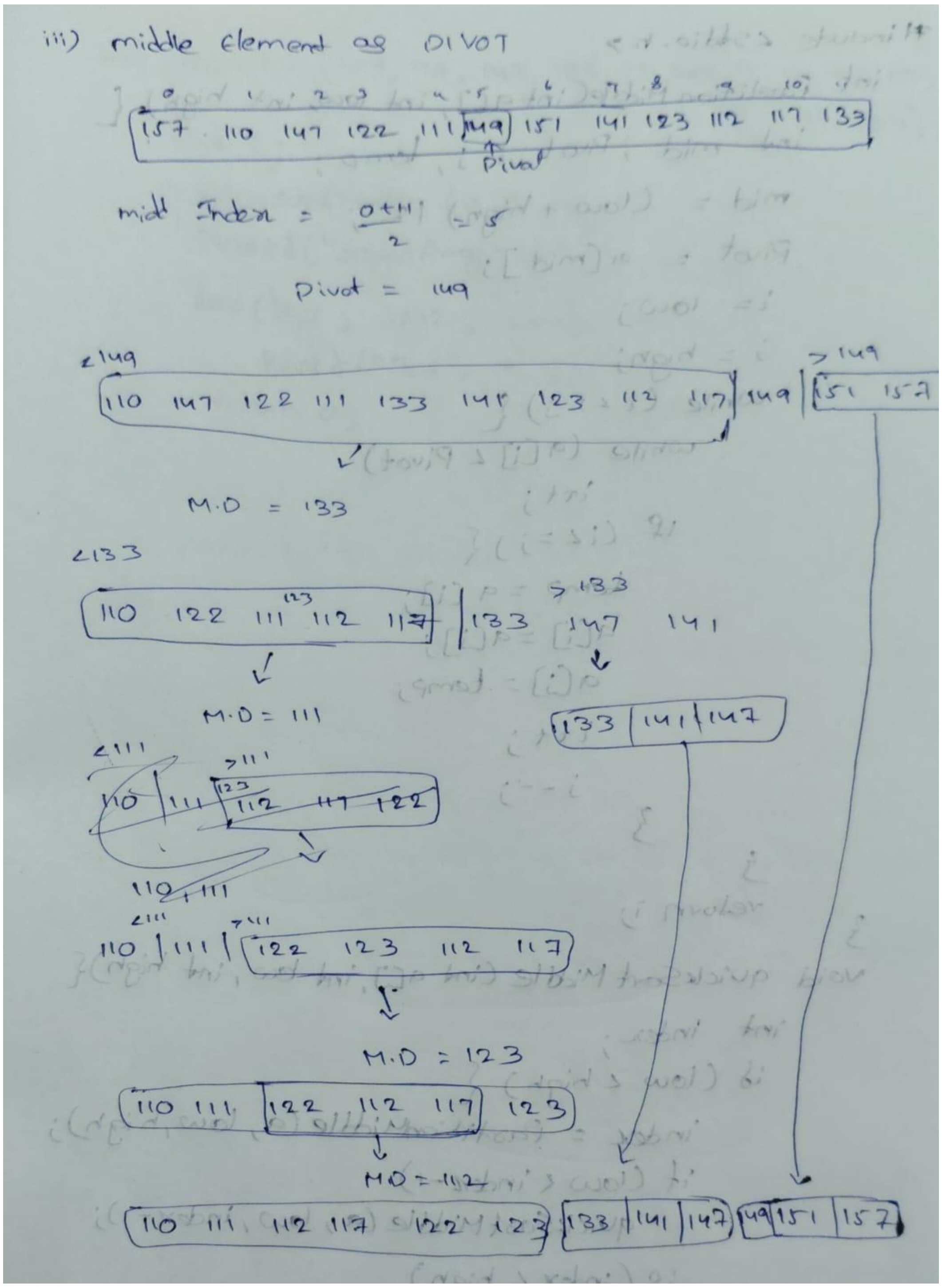
```
//CH.SC.U4CSE24121
#include <stdio.h>
int partitionMiddle(int a[], int low, int high)
{
    int mid, pivot, i, j, temp;
    mid = (low + high) / 2;
    pivot = a[mid];
    i = low;
    j = high;
    while (i <= j)
    {
        while (a[i] < pivot)
            i++;

        while (a[j] > pivot)
            j--;

        if (i <= j)
        {
            temp = a[i];
            a[i] = a[j];
            a[j] = temp;

            i++;
            j--;
        }
    }
    return i;
}
void quickSortMiddle(int a[], int low, int high)
{
    int index;
    if (low < high)
    {
        index = partitionMiddle(a, low, high);
        if (low < index - 1)
            quickSortMiddle(a, low, index - 1);
        if (index < high)
            quickSortMiddle(a, index, high);
    }
}
int main()
{
    int a[12] = {157,110,147,122,111,149,151,141,123,112,117,133};
    int i;
    quickSortMiddle(a, 0, 11);
    printf("Sorted Array:\n");
    for (i = 0; i < 12; i++)
        printf("%d ", a[i]);
    return 0;
}
```

Output:



```
Sorted Array:
110 111 112 117 122 123 133 141 147 149 151 157
--------------------------------
```

Handwritten:



iii) middle element as PIVOT

0	1	2	3	4	5	6	7	8	9	10	
157	110	147	122	111	149	151	141	123	112	117	133

Pivot

mid Index = $\frac{0+11}{2}$ => 5

Pivot = 149

<149   >149

110 147 122 111 133 141 123 112 117 | 149 | 151 157

M.D = 133

<133   >133

110 122 111 112 117 | 133 147 141

133 | 141 | 147

M.D = 111

<111   >111

~~110 | 111 | 112 117 122~~

110, 111

<111   >111

110 | 111 | 122 123 112 117

M.D = 123

110 111 | 122 112 117 | 123

M.D = 112

110 111 112 117 122 123 | 133 | 141 | 147 | 149 | 151 | 157