화일처리 보고서
B-Tree 삭제

20181653 이강희

November 25, 2019

강의 슬라이드를 참조하였습니다.

1 출력 결과

output.txt

```
File Processing
B-Tree
=============
M = 3
Delete 66       1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 20 22 24 25 28 30 33 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55
                56 57 58 59 60 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99 100
Delete 10       1 2 3 4 5 6 7 8 9 11 12 13 14 15 16 17 18 20 22 24 25 28 30 33 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56
                57 58 59 60 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99 100
Delete 22       1 2 3 4 5 6 7 8 9 11 12 13 14 15 16 17 18 20 24 25 28 30 33 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57
                58 59 60 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99 100
Delete 30       1 2 3 4 5 6 7 8 9 11 12 13 14 15 16 17 18 20 24 25 28 33 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58
                59 60 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99 100
Delete 44       1 2 3 4 5 6 7 8 9 11 12 13 14 15 16 17 18 20 24 25 28 33 40 41 42 43 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59
                60 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99 100
Delete 55       1 2 3 4 5 6 7 8 9 11 12 13 14 15 16 17 18 20 24 25 28 33 40 41 42 43 45 46 47 48 49 50 51 52 53 54 56 57 58 59 60
                61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99 100
Delete 50       1 2 3 4 5 6 7 8 9 11 12 13 14 15 16 17 18 20 24 25 28 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 60 61
                62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99 100
Delete 60       1 2 3 4 5 6 7 8 9 11 12 13 14 15 16 17 18 20 24 25 28 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62
                63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99 100
Delete 100      1 2 3 4 5 6 7 8 9 11 12 13 14 15 16 17 18 20 24 25 28 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62
                63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99
Delete 28       1 2 3 4 5 6 7 8 9 11 12 13 14 15 16 17 18 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63
                64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99
Delete 18       1 2 3 4 5 6 7 8 9 11 12 13 14 15 16 17 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64
                65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99
Delete 9        1 2 3 4 5 6 7 8 11 12 13 14 15 16 17 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64
                65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99
Delete 5        1 2 3 4 6 7 8 11 12 13 14 15 16 17 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65
                67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99
Delete 17       1 2 3 4 6 7 8 11 12 13 14 15 16 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67
                68 69 70 73 75 77 80 81 82 83 84 88 89 90 99
Delete 6        1 2 3 4 7 8 11 12 13 14 15 16 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68
                69 70 73 75 77 80 81 82 83 84 88 89 90 99
Delete 3        1 2 4 7 8 11 12 13 14 15 16 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68
                69 70 73 75 77 80 81 82 83 84 88 89 90 99
Delete 1        2 4 7 8 11 12 13 14 15 16 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69
                70 73 75 77 80 81 82 83 84 88 89 90 99
Delete 4        2 7 8 11 12 13 14 15 16 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70
                73 75 77 80 81 82 83 84 88 89 90 99
Delete 2        7 8 11 12 13 14 15 16 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70
                73 75 77 80 81 82 83 84 88 89 90 99
Delete 7        8 11 12 13 14 15 16 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 73
                75 77 80 81 82 83 84 88 89 90 99
Delete 8        11 12 13 14 15 16 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 73 75
                77 80 81 82 83 84 88 89 90 99
Delete 73       11 12 13 14 15 16 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 77
                80 81 82 83 84 88 89 90 99
Delete 12       11 13 14 15 16 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 77 80
                81 82 83 84 88 89 90 99
Delete 13       11 14 15 16 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 77 80 81
                82 83 84 88 89 90 99
Delete 14       11 15 16 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 77 80 81 82
                83 84 88 89 90 99
Delete 16       11 15 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 77 80 81 82 83
                84 88 89 90 99
Delete 15       11 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 77 80 81 82 83 84
                88 89 90 99
Delete 25       11 20 24 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 77 80 81 82 83 84 88
                89 90 99
Delete 24       11 20 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 77 80 81 82 83 84 88 89
                90 99
Delete 28       11 20 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 77 80 81 82 83 84 88 89
                90 99
Delete 40       11 20 33 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 77 80 81 82 83 84 88 89 90
                99
Delete 11       20 33 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 77 80 81 82 83 84 88 89 90 99
Delete 77       20 33 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 80 81 82 83 84 88 89 90 99
Delete 33       20 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 80 81 82 83 84 88 89 90 99
Delete 20       41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 80 81 82 83 84 88 89 90 99
Delete 90       41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 80 81 82 83 84 88 89 99
Delete 99       41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 80 81 82 83 84 88 89
Delete 70       41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 75 80 81 82 83 84 88 89
Delete 88       41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 75 80 81 82 83 84 89
Delete 80       41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 75 81 82 83 84 89
Delete 45       41 42 43 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 75 81 82 83 84 89
Delete 49       41 42 43 46 47 48 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 75 81 82 83 84 89
Delete 42       41 43 46 47 48 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 75 81 82 83 84 89
Delete 43       41 46 47 48 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 75 81 82 83 84 89
Delete 41       46 47 48 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 75 81 82 83 84 89
Delete 47       46 48 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 75 81 82 83 84 89
Delete 48       46 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 75 81 82 83 84 89
Delete 46       51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 75 81 82 83 84 89
Delete 63       51 52 53 54 56 57 58 59 61 62 64 65 67 68 69 75 81 82 83 84 89
Delete 68       51 52 53 54 56 57 58 59 61 62 64 65 67 69 75 81 82 83 84 89
Delete 53       51 52 54 56 57 58 59 61 62 64 65 67 69 75 81 82 83 84 89
Delete 57       51 52 54 56 58 59 61 62 64 65 67 69 75 81 82 83 84 89
Delete 52       51 54 56 58 59 61 62 64 65 67 69 75 81 82 83 84 89
Delete 56       51 54 58 59 61 62 64 65 67 69 75 81 82 83 84 89
Delete 83       51 54 58 59 61 62 64 65 67 69 75 81 82 84 89
Delete 81       51 54 58 59 61 62 64 65 67 69 75 82 84 89
Delete 82       51 54 58 59 61 62 64 65 67 69 75 84 89
Delete 84       51 54 58 59 61 62 64 65 67 69 75 89
Delete 75       51 54 58 59 61 62 64 65 67 69 89
Delete 89       51 54 58 59 61 62 64 65 67 69
Delete 61       51 54 58 59 62 64 65 67 69
Delete 62       51 54 58 59 64 65 67 69
Delete 64       51 54 58 59 65 67 69
Delete 69       51 54 58 59 65 67
Delete 67       51 54 58 59 65
Delete 65       51 54 58 59
Delete 54       51 58 59
Delete 59       51 58
Delete 58       51
Delete 51
=============
```

M = 4

Delete 66	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 20 22 24 25 28 30 33 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99 100
Delete 10	1 2 3 4 5 6 7 8 9 11 12 13 14 15 16 17 18 20 22 24 25 28 30 33 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99 100
Delete 22	1 2 3 4 5 6 7 8 9 11 12 13 14 15 16 17 18 20 24 25 28 30 33 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99 100
Delete 30	1 2 3 4 5 6 7 8 9 11 12 13 14 15 16 17 18 20 24 25 28 33 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99 100
Delete 44	1 2 3 4 5 6 7 8 9 11 12 13 14 15 16 17 18 20 24 25 28 33 40 41 42 43 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99 100
Delete 55	1 2 3 4 5 6 7 8 9 11 12 13 14 15 16 17 18 20 24 25 28 33 40 41 42 43 45 46 47 48 49 50 51 52 53 54 56 57 58 59 60 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99 100
Delete 50	1 2 3 4 5 6 7 8 9 11 12 13 14 15 16 17 18 20 24 25 28 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 60 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99 100
Delete 60	1 2 3 4 5 6 7 8 9 11 12 13 14 15 16 17 18 20 24 25 28 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99 100
Delete 100	1 2 3 4 5 6 7 8 9 11 12 13 14 15 16 17 18 20 24 25 28 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99
Delete 28	1 2 3 4 5 6 7 8 9 11 12 13 14 15 16 17 18 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99
Delete 18	1 2 3 4 5 6 7 8 9 11 12 13 14 15 16 17 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99
Delete 9	1 2 3 4 5 6 7 8 11 12 13 14 15 16 17 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99
Delete 5	1 2 3 4 6 7 8 11 12 13 14 15 16 17 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99
Delete 17	1 2 3 4 6 7 8 11 12 13 14 15 16 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99
Delete 6	1 2 3 4 7 8 11 12 13 14 15 16 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99
Delete 3	1 2 4 7 8 11 12 13 14 15 16 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99
Delete 1	2 4 7 8 11 12 13 14 15 16 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99
Delete 4	2 7 8 11 12 13 14 15 16 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99
Delete 2	7 8 11 12 13 14 15 16 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99
Delete 7	8 11 12 13 14 15 16 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99
Delete 8	11 12 13 14 15 16 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 73 75 77 80 81 82 83 84 88 89 90 99
Delete 73	11 12 13 14 15 16 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 77 80 81 82 83 84 88 89 90 99
Delete 12	11 13 14 15 16 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 77 80 81 82 83 84 88 89 90 99
Delete 13	11 14 15 16 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 77 80 81 82 83 84 88 89 90 99
Delete 14	11 15 16 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 77 80 81 82 83 84 88 89 90 99
Delete 16	11 15 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 77 80 81 82 83 84 88 89 90 99
Delete 15	11 20 24 25 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 77 80 81 82 83 84 88 89 90 99
Delete 25	11 20 24 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 77 80 81 82 83 84 88 89 90 99
Delete 24	11 20 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 77 80 81 82 83 84 88 89 90 99
Delete 28	11 20 33 40 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 77 80 81 82 83 84 88 89 90 99
Delete 40	11 20 33 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 77 80 81 82 83 84 88 89 90 99
Delete 11	20 33 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 77 80 81 82 83 84 88 89 90 99
Delete 77	20 33 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 80 81 82 83 84 88 89 90 99
Delete 33	20 41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 80 81 82 83 84 88 89 90 99
Delete 20	41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 80 81 82 83 84 88 89 90 99
Delete 90	41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 80 81 82 83 84 88 89 99
Delete 99	41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 70 75 80 81 82 83 84 88 89
Delete 70	41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 75 80 81 82 83 84 88 89
Delete 88	41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 75 80 81 82 83 84 89
Delete 80	41 42 43 45 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 75 81 82 83 84 89
Delete 45	41 42 43 46 47 48 49 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 75 81 82 83 84 89
Delete 49	41 42 43 46 47 48 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 75 81 82 83 84 89
Delete 42	41 43 46 47 48 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 75 81 82 83 84 89
Delete 43	41 46 47 48 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 75 81 82 83 84 89
Delete 41	46 47 48 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 75 81 82 83 84 89
Delete 47	46 48 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 75 81 82 83 84 89
Delete 48	46 51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 75 81 82 83 84 89
Delete 46	51 52 53 54 56 57 58 59 61 62 63 64 65 67 68 69 75 81 82 83 84 89
Delete 63	51 52 53 54 56 57 58 59 61 62 64 65 67 68 69 75 81 82 83 84 89
Delete 68	51 52 53 54 56 57 58 59 61 62 64 65 67 69 75 81 82 83 84 89
Delete 53	51 52 54 56 57 58 59 61 62 64 65 67 69 75 81 82 83 84 89
Delete 57	51 52 54 56 58 59 61 62 64 65 67 69 75 81 82 83 84 89
Delete 52	51 54 56 58 59 61 62 64 65 67 69 75 81 82 83 84 89
Delete 56	51 54 58 59 61 62 64 65 67 69 75 81 82 83 84 89
Delete 83	51 54 58 59 61 62 64 65 67 69 75 81 82 84 89
Delete 81	51 54 58 59 61 62 64 65 67 69 75 82 84 89
Delete 82	51 54 58 59 61 62 64 65 67 69 75 84 89
Delete 84	51 54 58 59 61 62 64 65 67 69 75 89
Delete 75	51 54 58 59 61 62 64 65 67 69 89
Delete 89	51 54 58 59 61 62 64 65 67 69
Delete 61	51 54 58 59 62 64 65 67 69
Delete 62	51 54 58 59 64 65 67 69
Delete 64	51 54 58 59 65 67 69
Delete 69	51 54 58 59 65 67
Delete 67	51 54 58 59 65
Delete 65	51 54 58 59
Delete 54	51 58 59
Delete 59	51 58
Delete 58	51
Delete 51	

2 소스 코드

```cpp
#include <iostream>
#include <vector>
#include <stack>
#include <algorithm>
using namespace std;

enum FLAG {LEFT, RIGHT, MERGE};

class BTreeNode {
public:
    BTreeNode(int key) {
        keys.push_back(key);
        childs.assign(2, nullptr);
        n = 1;
    }
    BTreeNode(const BTreeNode *node) : n(node->n), keys(node->keys), childs(node->childs) {}
    ~BTreeNode() {
        for (auto child : childs) delete child;
    }
    void insertKey(int newKey, BTreeNode *newChild=nullptr, int newChildPos=RIGHT) {
        keys.push_back(newKey);
        childs.push_back(newChild);
        int i = ++n - 1;
        while (i > 0 && keys[i] < keys[i-1]) {
            swap(keys[i], keys[i-1]);
            swap(childs[i], childs[i+1]);
            i--;
        }
        if (newChildPos == LEFT) swap(childs[i], childs[i+1]);
    }

    bool deleteKey(int oldKey, int oldChildPos=RIGHT) {
        int i = 0;
        while (i < n && oldKey > keys[i]) i++;
        if (keys[i] != oldKey) return false;
        keys.erase(keys.begin() + i);
        if (oldChildPos == RIGHT) i++;
        childs.erase(childs.begin() + i);
        n--;
        return true;
    }

    int n = 0;
    vector<int> keys;
    vector<BTreeNode*> childs;
    static int M;
};

int BTreeNode::M = 3;

bool insertBT(BTreeNode *&T, int newKey) {
    if (T == nullptr) {
        T = new BTreeNode(newKey);
        return true;
```

```
55      }
56      stack<BTreeNode*> st;
57      BTreeNode *x = T;
58      while (x != nullptr) {
59          int i = 0;
60          while (i < x->n && newKey > x->keys[i]) i++;
61          if (i < x->n && newKey == x->keys[i]) return false;
62          st.push(x);
63          x = x->childs[i];
64      }
65
66      BTreeNode *newChild = nullptr;
67      while (true) {
68          x = st.top(); st.pop();
69          x->insertKey(newKey, newChild);
70          if (x->n < BTreeNode::M) break;
71          newKey = x->keys[x->n/2];
72          newChild = new BTreeNode(x);
73          x->keys.resize(x->n/2);
74          x->childs.resize(x->n/2+1);
75          x->n /= 2;
76          newChild->keys.erase(newChild->keys.begin(), newChild->keys.begin() + newChild->n/2+1);
77          newChild->childs.erase(newChild->childs.begin(), newChild->childs.begin() + newChild->n/2+1);
78          newChild->n -= newChild->n / 2 + 1;
79          if (st.empty()) {
80              T = new BTreeNode(newKey);
81              T->childs[0] = x;
82              T->childs[1] = newChild;
83              break;
84          }
85      }
86
87      return true;
88  }
89
90  bool deleteBT(BTreeNode *&T, int oldKey) {
91      stack<BTreeNode*> st;
92      BTreeNode *x = T;
93      while (x != nullptr) {
94          int i = 0;
95          while (i < x->n && oldKey > x->keys[i]) i++;
96          if (i < x->n && oldKey == x->keys[i]) break;
97          st.push(x);
98          x = x->childs[i];
99      }
100     if (x == nullptr) return false; // Search Fail
101     if (x->childs[0] != nullptr) { // if x is Internal Node
102         BTreeNode *internalNode = x;
103         st.push(x);
104         x = x->childs[find(x->keys.begin(), x->keys.end(), oldKey) - x->keys.begin()+1];
105         while (x != nullptr) {
106             st.push(x);
107             x = x->childs[0];
108         }
109         x = st.top(); st.pop();
110         *find(internalNode->keys.begin(), internalNode->keys.end(), oldKey) = x->keys[0];
```

```
            x->keys[0] = oldKey;
        }
        x->deleteKey(oldKey);
        BTreeNode *y, *leftSibling, *rightSibling;
        while (x->n < (BTreeNode::M+1)/2-1 && !st.empty()) { // While Underflow
            y = st.top(); st.pop();
            int idx = find(y->childs.begin(), y->childs.end(), x) - y->childs.begin();
            if (x == y->childs[0]) leftSibling = nullptr;
            else leftSibling = y->childs[idx-1];
            if (x == y->childs.back()) rightSibling = nullptr;
            else rightSibling = y->childs[idx+1];
            int flag;
            if (leftSibling == nullptr || rightSibling == nullptr) {
                if (leftSibling == nullptr) {
                    if (rightSibling->n > (BTreeNode::M-1)/2) flag = RIGHT;
                    else flag = MERGE;
                }
                else {
                    if (leftSibling->n > (BTreeNode::M-1)/2) flag = LEFT;
                    else flag = MERGE;
                }
            }
            else {
                if (leftSibling->n >= rightSibling->n && leftSibling->n > (BTreeNode::M-1)/2) flag = LEFT;
                else if (rightSibling->n > leftSibling->n) flag = RIGHT;
                else flag = MERGE;
            }
            if (flag == LEFT) { // Get key from left sibling
                x->insertKey(y->keys[idx-1], leftSibling->childs.back(), LEFT);
                y->keys[idx-1] = leftSibling->keys.back();
                leftSibling->deleteKey(leftSibling->keys.back());
            }
            else if (flag == RIGHT) { // Get key from right sibling
                x->insertKey(y->keys[idx], rightSibling->childs[0], RIGHT);
                y->keys[idx] = rightSibling->keys[0];
                rightSibling->deleteKey(rightSibling->keys[0], LEFT);
            }
            else { // Merge
                if (leftSibling == nullptr) {
                    idx++;
                    leftSibling = x;
                    x = rightSibling;
                }
                leftSibling->keys.push_back(y->keys[idx-1]);
                leftSibling->n++;
                y->deleteKey(y->keys[idx-1], RIGHT);
                leftSibling->keys.insert(leftSibling->keys.end(), x->keys.begin(), x->keys.end());
                leftSibling->childs.insert(leftSibling->childs.end(), x->childs.begin(), x->childs.end());
                leftSibling->n += x->n;
                x->childs.clear();
                delete x;
            }
            x = y;
        }
        if (T->n == 0) {
            y = T;
```

```
167             T = T->childs[0];
168             y->childs.clear();
169             delete y;
170         }
171 
172         return true;
173     }
174 
175     void inorderBT(BTreeNode *T) {
176         if (T == nullptr) return;
177         for (int i=0;i<T->n;++i) {
178             inorderBT(T->childs[i]);
179             cout << T->keys[i] << ' ';
180         }
181         inorderBT(T->childs[T->n]);
182     }
183 
184     int main() {
185         cout << " File Processing  " << endl;
186         cout << "      B-Tree" << endl;
187         cout << "==================" << endl;
188 
189         int insertKeys[] = {
190             40, 11, 77, 33, 20, 90, 99, 70, 88, 80,
191             66, 10, 22, 30, 44, 55, 50, 60, 100, 28,
192             18, 9, 5, 17, 6, 3, 1, 4, 2, 7,
193             8, 73, 12, 13, 14, 16, 15, 25, 24, 28,
194             45, 49, 42, 43, 41, 47, 48, 46, 63, 68,
195             61, 62, 64, 69, 67, 65, 54, 59, 58, 51,
196             53, 57, 52, 56, 83, 81, 82, 84, 75, 89
197         };
198 
199         int deleteKeys[] = {
200             66, 10, 22, 30, 44, 55, 50, 60, 100, 28,
201             18, 9, 5, 17, 6, 3, 1, 4, 2, 7,
202             8, 73, 12, 13, 14, 16, 15, 25, 24, 28,
203             40, 11, 77, 33, 20, 90, 99, 70, 88, 80,
204             45, 49, 42, 43, 41, 47, 48, 46, 63, 68,
205             53, 57, 52, 56, 83, 81, 82, 84, 75, 89,
206             61, 62, 64, 69, 67, 65, 54, 59, 58, 51
207         };
208 
209         BTreeNode *root = nullptr;
210 
211         BTreeNode::M = 3;
212         cout << "M = 3" << endl;
213         for (int key : insertKeys) insertBT(root, key);
214 
215         for (int key : deleteKeys) {
216             cout << "Delete " << key << '\t';
217             deleteBT(root, key);
218             inorderBT(root);
219             cout << endl;
220         }
221 
222         delete root;
```

```
223     root = nullptr;
224 
225     cout << "==================" << endl;
226     BTreeNode::M = 4;
227     cout << "M = 4" << endl;
228     for (int key : insertKeys) insertBT(root, key);
229 
230     for (int key : deleteKeys) {
231         cout << "Delete " << key << '\t';
232         deleteBT(root, key);
233         inorderBT(root);
234         cout << endl;
235     }
236 
237     delete root;
238 
239     return 0;
240 }
```