

Aufgabenblatt 9: Reihungen (2)

Florian Ludewig (Übungsgruppe 2)

20. Dezember 2019

Aufgabe 1 – Binäre Suche

a)

```
1 int linearSearch(int n, int a[], int l) {
2     for (int i = 0; i < l; i++) {
3         if (a[i] == n) return i;
4     }
5     return -1;
6 }
7
8 int binarySearch(int n, int a[], int l) {
9     int range_start = 0, range_end = l - 1;
10    while (1) {
11        int middle = (range_start + range_end) / 2;
12        if (a[middle] == n) return middle;
13        if (range_start > range_end) return -1;
14        if (a[middle] < n) range_start = middle + 1;
15        else range_end = middle - 1;
16    }
17 }
```

b)

```
1 int main(void) {
2     int n;
3     printf("Zahl eingeben: ");
4     scanf("%d", &n);
5
6     int primes[25] = {2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47,
7         53, 59, 61, 67, 71, 73, 79, 83, 89, 97};
8     int position_linear_search = linearSearch(n, primes, 25);
9     int position_binary_search = binarySearch(n, primes, 25);
10
11    printf("Suchergebnis der linearen Suche: %d\n", position_linear_search);
12    printf("Suchergebnis der binaeren Suche: %d\n", position_binary_search);
13
14 }
```

c)

```
1 int binarySearch(int n, int a[], int l) {
2     int* range_start = &a[0];
3     int* range_end = &a[l - 1];
4     while(1) {
5         if (range_start > range_end) return -1;
6         int* middle = ((range_end - range_start) / 2) + range_start;
7         if (*middle == n) return middle - &a[0];
8         if (*middle > n) range_end = middle - 1;
9         else range_start = middle + 1;
10    }
11 }
```

Aufgabe 2 – Magisches Quadrat

```
1 #include<stdio.h>
2
3 void generate_magic_square(int n) {
4     int square[n][n];
5
6     for (int i = 0; i < n; i++)
7         for (int j = 0; j < n; j++)
8             square[i][j] = 0;
9
10    int row = n - 1, column = n / 2;
11    int successor = 2;
12    square[row][column] = 1;
13
14    while(successor <= n * n) {
15        row = row == n - 1 ? 0 : row + 1;
16        column = column == n - 1 ? 0 : column + 1;
17        if (square[row][column] == 0) {
18            square[row][column] = successor;
19            successor++;
20        } else {
21            while (1) {
22                row = row == n - 1 ? 0 : row + 1;
23                column = column == 0 ? n - 1 : column - 1;
24                if (square[row][column] == 0) {
25                    square[row][column] = successor;
26                    successor++;
27                    break;
28                }
29            }
30        }
31    }
32
33    for (int i = 0; i < n; i++) {
34        for (int j = 0; j < n; j++)
35            printf("%d ", square[i][j]);
36        printf("\n");
37    }
38    printf("\n");
39 }
40
41 int main(void) {
42     int n;
43     printf("Ungerade Zahl eingeben: ");
44     scanf("%d", &n);
45     generate_magic_square(n);
46     return 0;
47 }
```