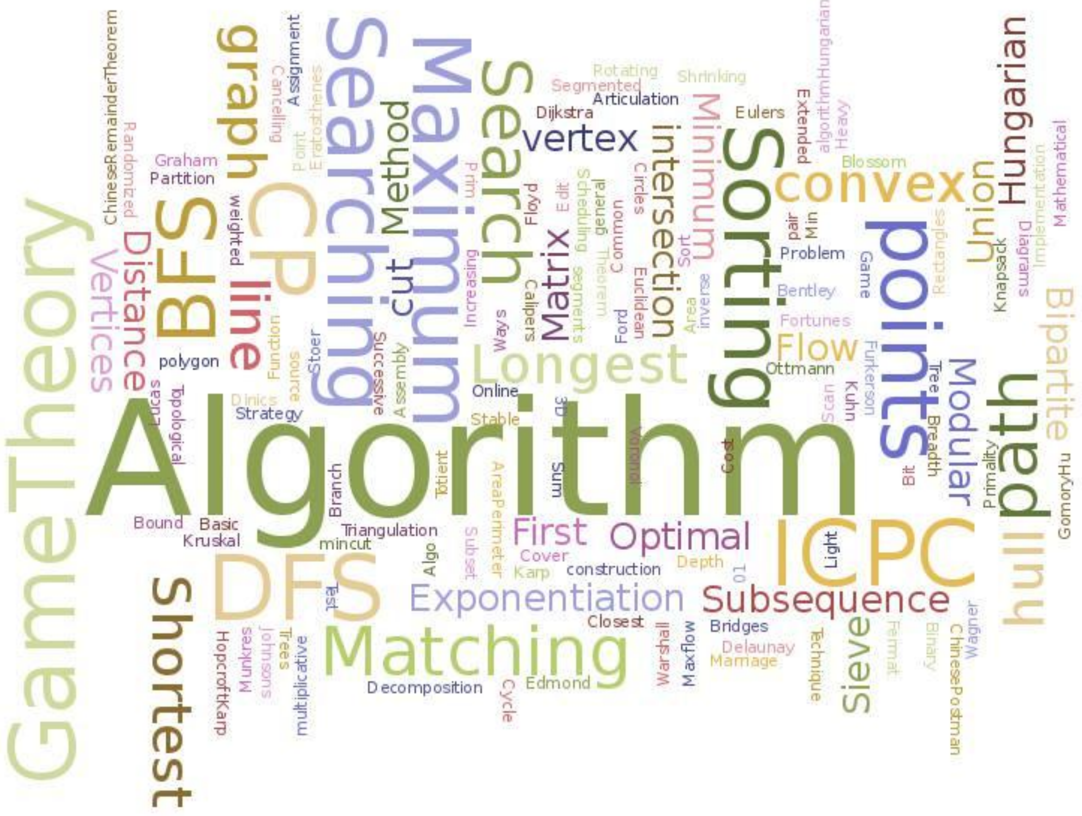
AMRITA VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY UNDER SECTION 3 OF UGC ACT, 1956

SCHOOL OF COMPUTING

Design and Analysis of Algorithms Lab 1 Report

(Course Code: 23CSE211)



Name: NAGARAMPALLI SARVAN KUMAR

Roll Number: CH.SC.U4CSE24130

Branch/ Section: Computer Science and Engineering / CSE – B

College: Amrita Vishwa Vidyapeetham Chennai Campus

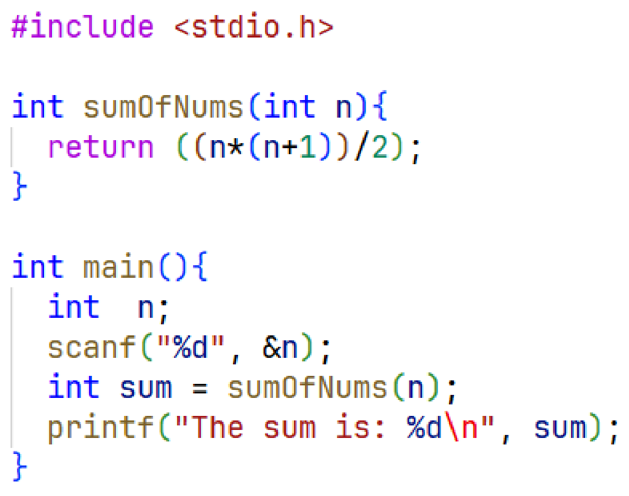
Academic Year: 2025 – 2026

Programs and Their Space Complexities Date: 27-11-2025

1. Write a program to find the sum of first *n* natural numbers using a function.
2. Write a program to find the sum of squares of first *n* natural numbers.
3. Write a program to find the sum of cubes of first *n* natural numbers.
4. Write a program to find factorial of a number using a recursive function.
5. Write a program to find the transpose of a 3×3 matrix.
6. Write a program to print the Fibonacci series using recursion.

Solutions:

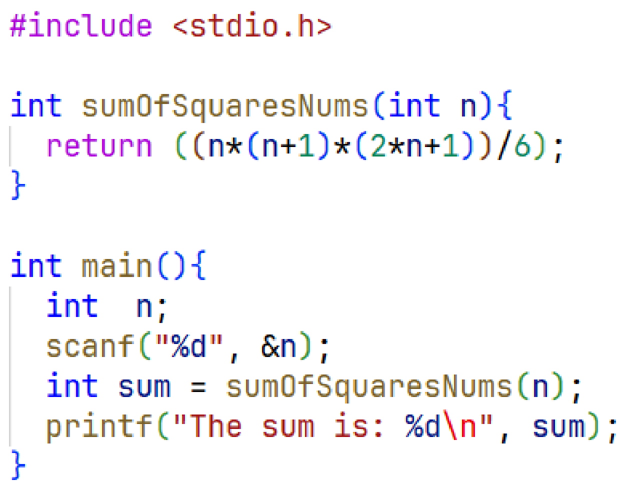
Write a program to find the sum of first *n* natural numbers using a function.



```c
#include <stdio.h>

int sumOfNums(int n){
  return ((n*(n+1))/2);
}

int main(){
  int  n;
  scanf("%d", &n);
  int sum = sumOfNums(n);
  printf("The sum is: %d\n", sum);
}
```

Justification: Uses the formula n(n+1)/2, which calculates the total directly.
Space Complexity: O(1)

Write a program to find the sum of squares of first *n* natural numbers.

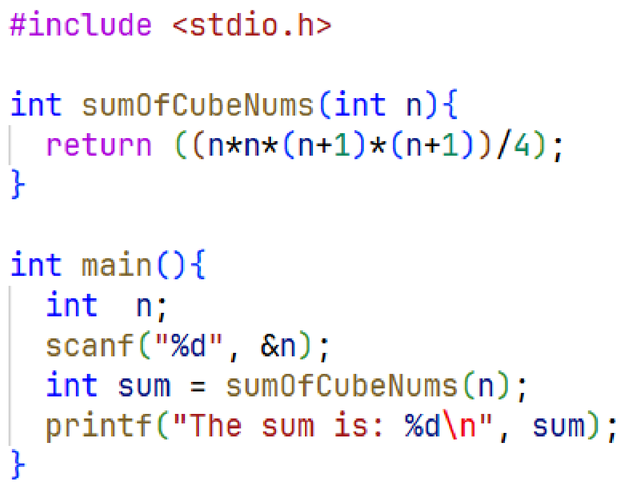


```c
#include <stdio.h>

int sumOfSquaresNums(int n){
  return ((n*(n+1)*(2*n+1))/6);
}

int main(){
  int  n;
  scanf("%d", &n);
  int sum = sumOfSquaresNums(n);
  printf("The sum is: %d\n", sum);
}
```

Justification: Uses the formula n(n+1)(2n+1)/6, computes the sum directly.

Space Complexity: O(1)

Write a program to find the sum of cubes of first *n* natural numbers.

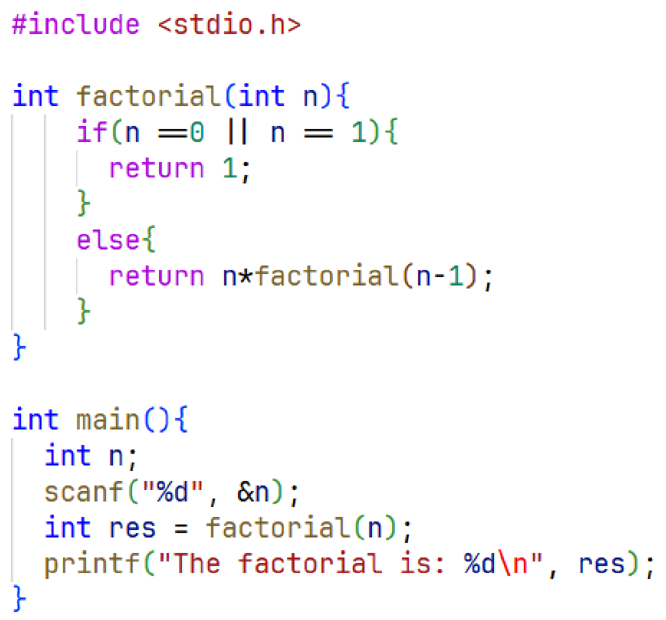


```c
#include <stdio.h>

int sumOfCubeNums(int n){
  return ((n*n*(n+1)*(n+1))/4);
}

int main(){
  int  n;
  scanf("%d", &n);
  int sum = sumOfCubeNums(n);
  printf("The sum is: %d\n", sum);
}
```

Justification: Uses the formula $[(n(n+1))/2]^2$, which calculates the total directly.

Space Complexity: O(1)

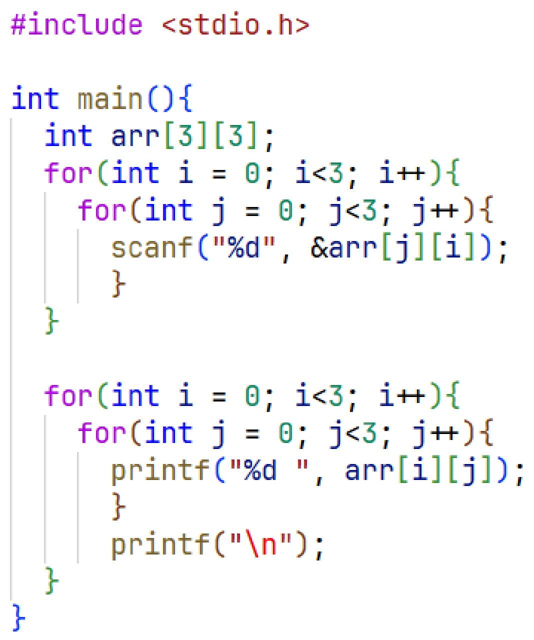
Write a program to find factorial of a number using a recursive function.



```c
#include <stdio.h>

int factorial(int n){
    if(n ==0 || n == 1){
        return 1;
    }
    else{
        return n*factorial(n-1);
    }
}

int main(){
    int n;
    scanf("%d", &n);
    int res = factorial(n);
    printf("The factorial is: %d\n", res);
}
```

Justification: Uses recursive calls that reduce n step-by-step until the base case is reached.
Space Complexity: O(n)

Write a program to find the transpose of a 3×3 matrix.

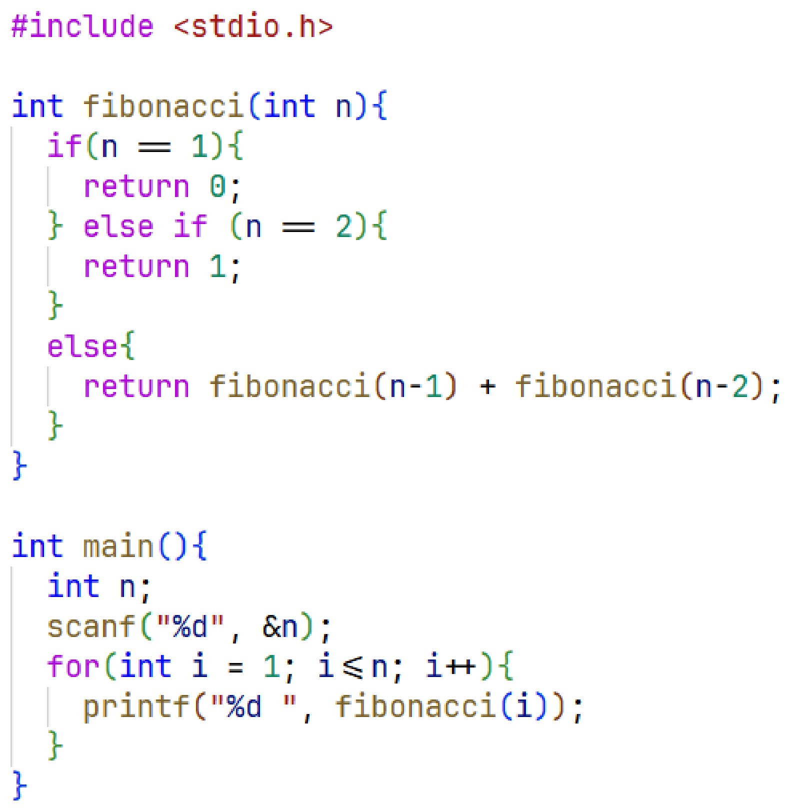


```c
#include <stdio.h>

int main(){
    int arr[3][3];
    for(int i = 0; i<3; i++){
        for(int j = 0; j<3; j++){
            scanf("%d", &arr[j][i]);
            }
    }

    for(int i = 0; i<3; i++){
        for(int j = 0; j<3; j++){
            printf("%d ", arr[i][j]);
            }
            printf("\n");
    }
}
```

Justification: Uses a fixed 3×3 array and stores each element directly in its transposed position during input.

Space Complexity: O(1)

Write a program to print the Fibonacci series using recursion.



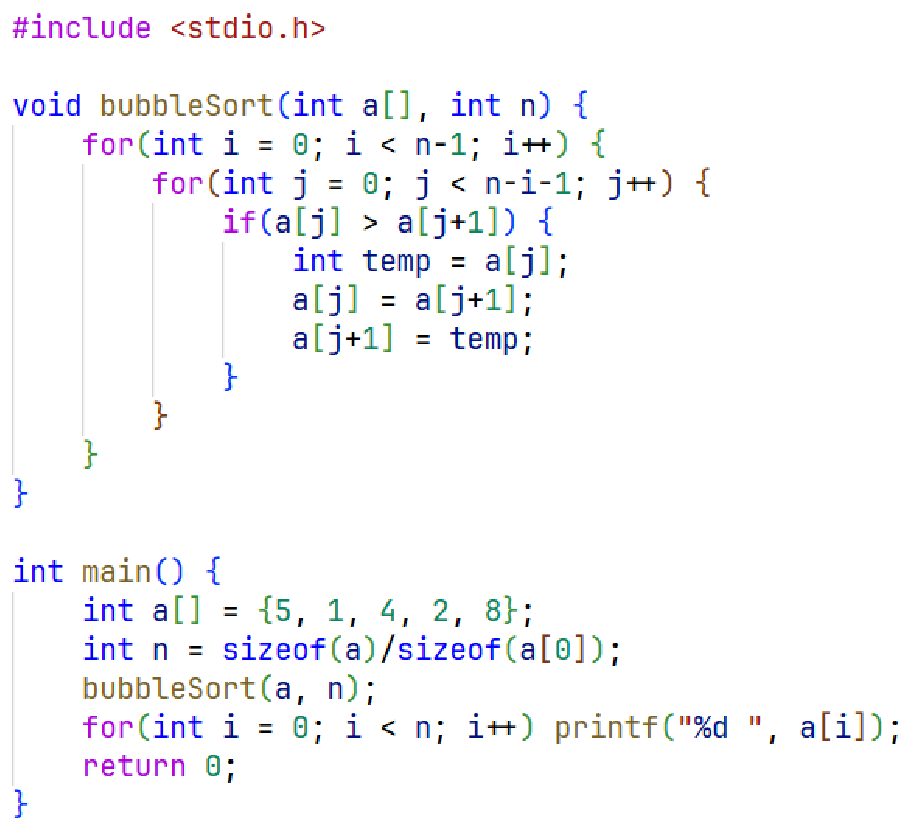
```c
#include <stdio.h>

int fibonacci(int n){
    if(n == 1){
        return 0;
    } else if (n == 2){
        return 1;
    }
    else{
        return fibonacci(n-1) + fibonacci(n-2);
    }
}

int main(){
    int n;
    scanf("%d", &n);
    for(int i = 1; i<=n; i++){
        printf("%d ", fibonacci(i));
    }
}
```

Justification: Uses recursive calls that expand until the base cases are reached.
Space Complexity: O(n)

Sorting Techniques Date: 04-12-2025

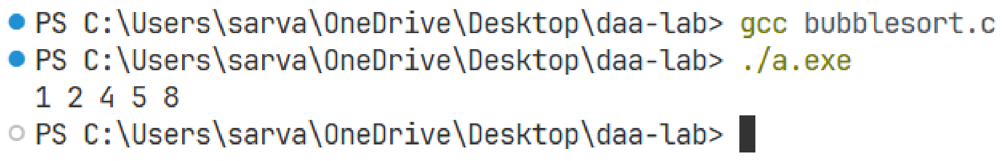
1. Bubble Sort in C
2. Insertion Sort in C
3. Selection Sort in C
4. Bucket Sort in C
5. Heap Sort in C (Max Heap)
6. Heap Sort in C (Min Heap)

Bubble Sort in C



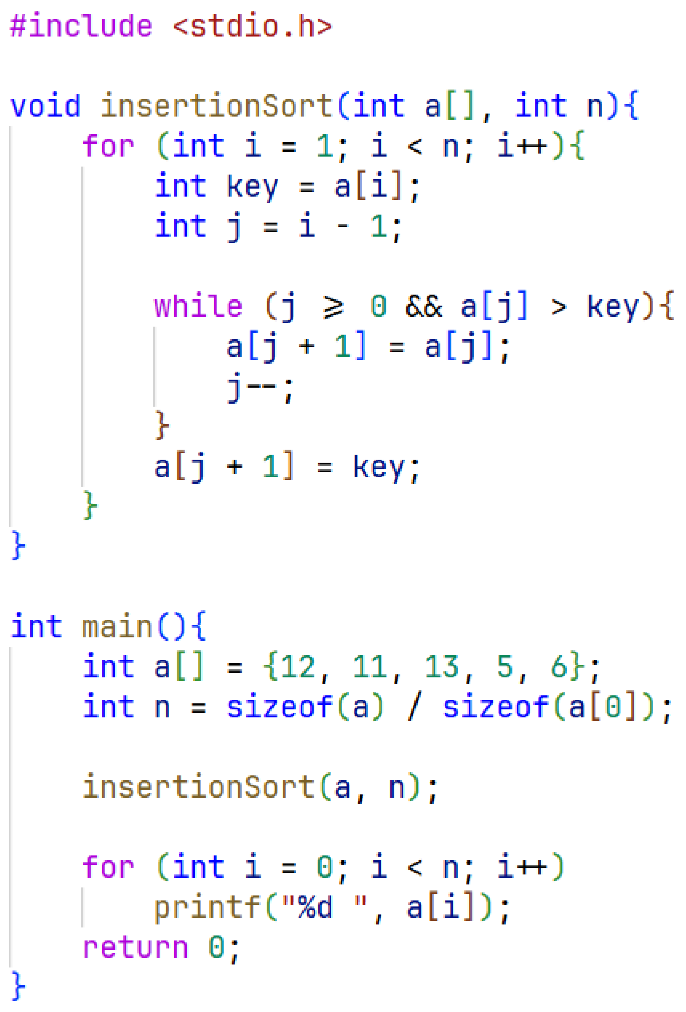
```c
#include <stdio.h>

void bubbleSort(int a[], int n) {
    for(int i = 0; i < n-1; i++) {
        for(int j = 0; j < n-i-1; j++) {
            if(a[j] > a[j+1]) {
                int temp = a[j];
                a[j] = a[j+1];
                a[j+1] = temp;
            }
        }
    }
}

int main() {
    int a[] = {5, 1, 4, 2, 8};
    int n = sizeof(a)/sizeof(a[0]);
    bubbleSort(a, n);
    for(int i = 0; i < n; i++) printf("%d ", a[i]);
    return 0;
}
```

Output:



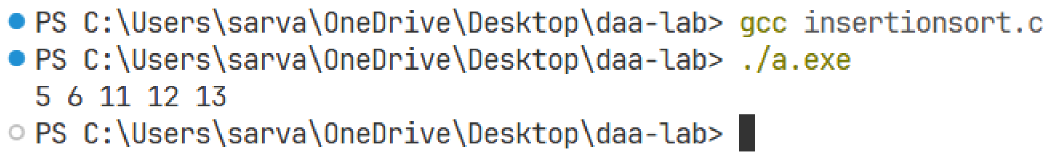
```
PS C:\Users\sarva\OneDrive\Desktop\daa-lab> gcc bubblesort.c
PS C:\Users\sarva\OneDrive\Desktop\daa-lab> ./a.exe
1 2 4 5 8
PS C:\Users\sarva\OneDrive\Desktop\daa-lab>
```

Insertion Sort in C



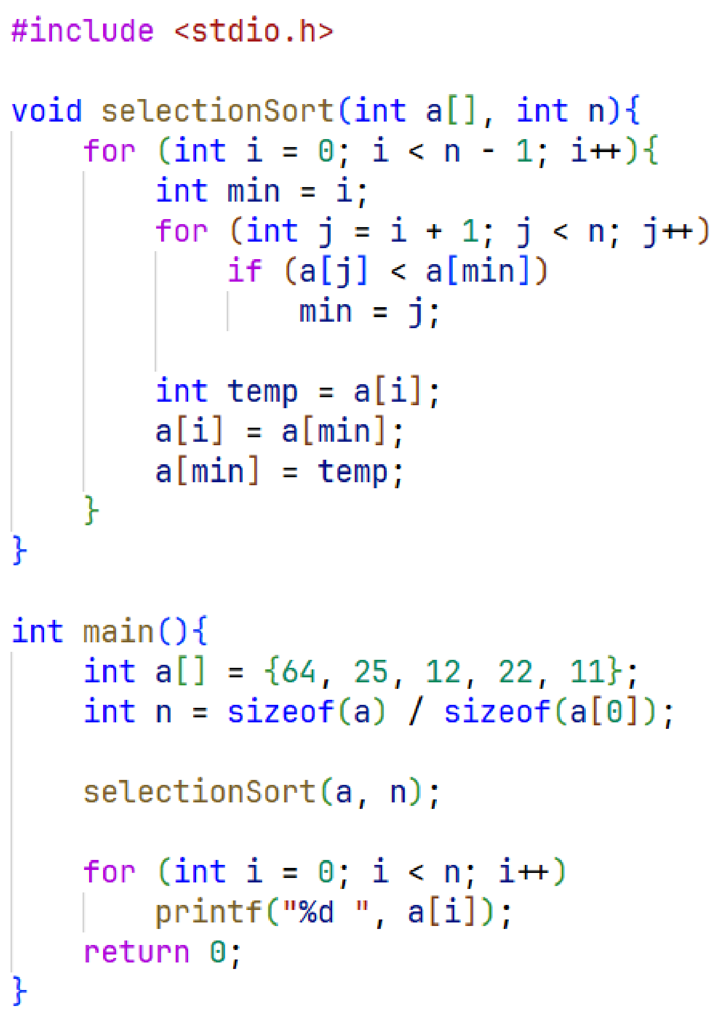
```c
#include <stdio.h>

void insertionSort(int a[], int n){
    for (int i = 1; i < n; i++){
        int key = a[i];
        int j = i - 1;

        while (j >= 0 && a[j] > key){
            a[j + 1] = a[j];
            j--;
        }
        a[j + 1] = key;
    }
}

int main(){
    int a[] = {12, 11, 13, 5, 6};
    int n = sizeof(a) / sizeof(a[0]);

    insertionSort(a, n);

    for (int i = 0; i < n; i++)
        printf("%d ", a[i]);
    return 0;
}
```

Output:



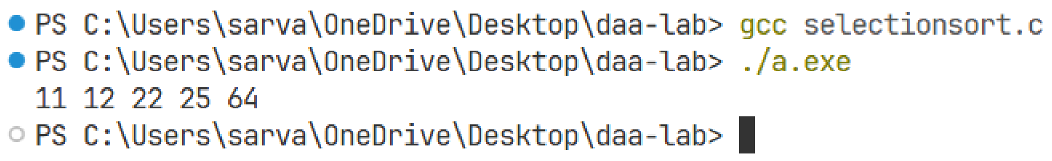
```
● PS C:\Users\sarva\OneDrive\Desktop\daa-lab> gcc insertionsort.c
● PS C:\Users\sarva\OneDrive\Desktop\daa-lab> ./a.exe
  5 6 11 12 13
○ PS C:\Users\sarva\OneDrive\Desktop\daa-lab> █
```

Selection Sort in C



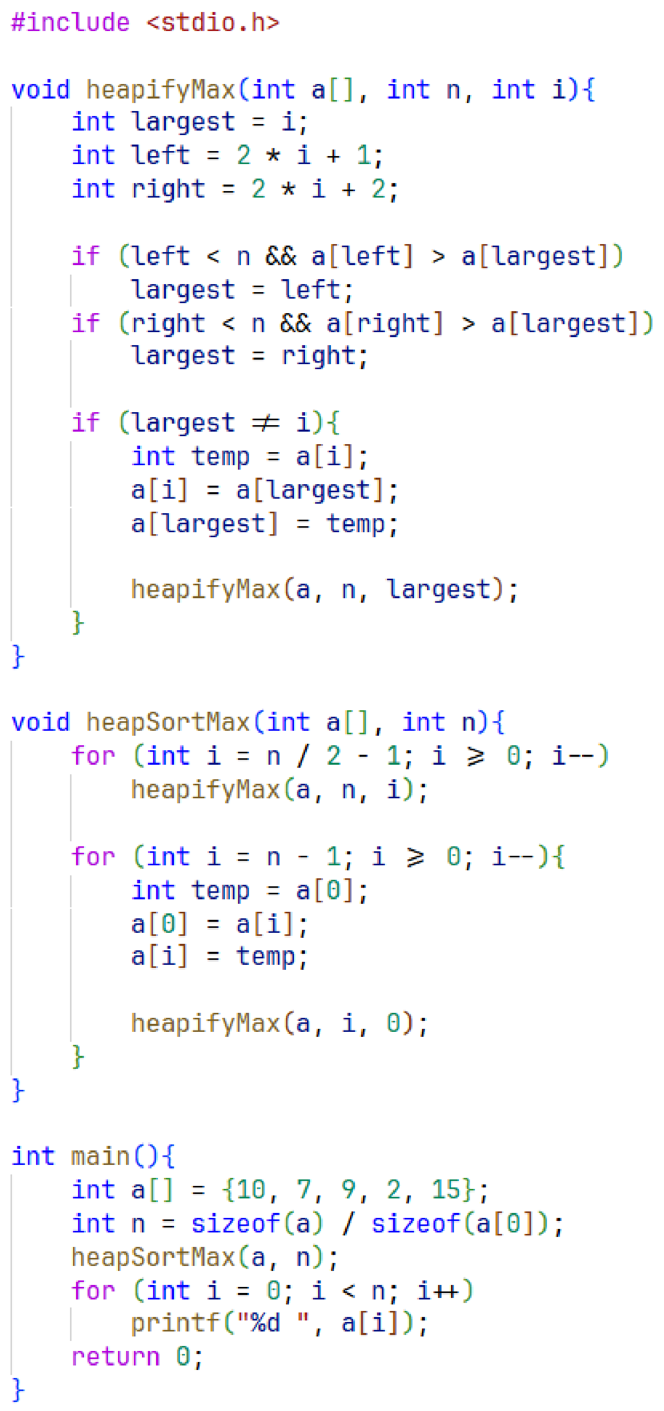
```c
#include <stdio.h>

void selectionSort(int a[], int n){
    for (int i = 0; i < n - 1; i++){
        int min = i;
        for (int j = i + 1; j < n; j++)
            if (a[j] < a[min])
                min = j;

        int temp = a[i];
        a[i] = a[min];
        a[min] = temp;
    }
}

int main(){
    int a[] = {64, 25, 12, 22, 11};
    int n = sizeof(a) / sizeof(a[0]);

    selectionSort(a, n);

    for (int i = 0; i < n; i++)
        printf("%d ", a[i]);
    return 0;
}
```

Output:



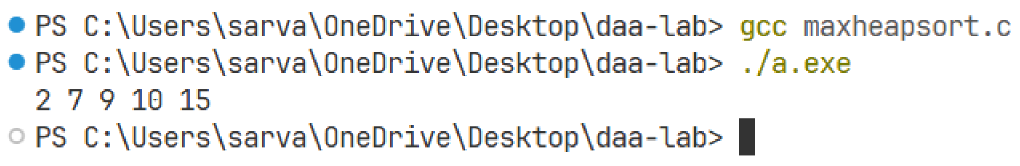
```
PS C:\Users\sarva\OneDrive\Desktop\daa-lab> gcc selectionsort.c
PS C:\Users\sarva\OneDrive\Desktop\daa-lab> ./a.exe
11 12 22 25 64
PS C:\Users\sarva\OneDrive\Desktop\daa-lab>
```

Heap Sort using Max Heap in C



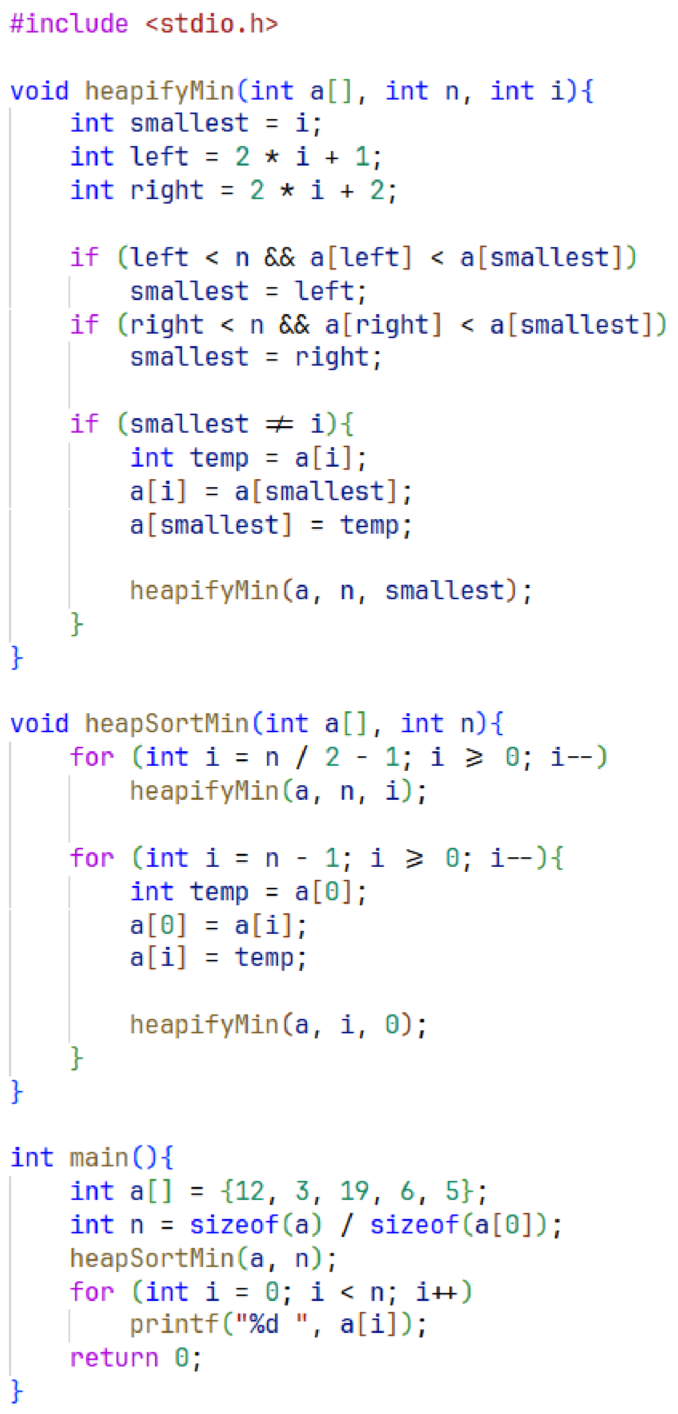
```c
#include <stdio.h>

void heapifyMax(int a[], int n, int i){
    int largest = i;
    int left = 2 * i + 1;
    int right = 2 * i + 2;

    if (left < n && a[left] > a[largest])
        largest = left;
    if (right < n && a[right] > a[largest])
        largest = right;

    if (largest != i){
        int temp = a[i];
        a[i] = a[largest];
        a[largest] = temp;

        heapifyMax(a, n, largest);
    }
}

void heapSortMax(int a[], int n){
    for (int i = n / 2 - 1; i >= 0; i--)
        heapifyMax(a, n, i);

    for (int i = n - 1; i >= 0; i--){
        int temp = a[0];
        a[0] = a[i];
        a[i] = temp;

        heapifyMax(a, i, 0);
    }
}

int main(){
    int a[] = {10, 7, 9, 2, 15};
    int n = sizeof(a) / sizeof(a[0]);
    heapSortMax(a, n);
    for (int i = 0; i < n; i++)
        printf("%d ", a[i]);
    return 0;
}
```

Output:



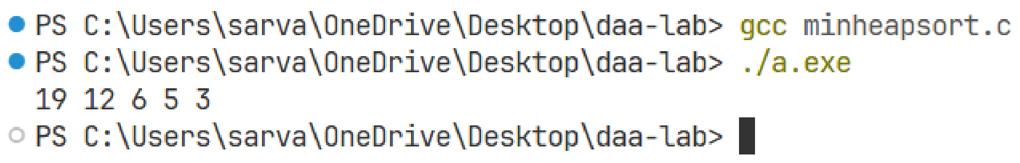
```
● PS C:\Users\sarva\OneDrive\Desktop\daa-lab> gcc maxheapsort.c
● PS C:\Users\sarva\OneDrive\Desktop\daa-lab> ./a.exe
  2 7 9 10 15
○ PS C:\Users\sarva\OneDrive\Desktop\daa-lab> 
```

Heap Sort using Min Heap in C



```c
#include <stdio.h>

void heapifyMin(int a[], int n, int i){
    int smallest = i;
    int left = 2 * i + 1;
    int right = 2 * i + 2;

    if (left < n && a[left] < a[smallest])
        smallest = left;
    if (right < n && a[right] < a[smallest])
        smallest = right;

    if (smallest != i){
        int temp = a[i];
        a[i] = a[smallest];
        a[smallest] = temp;

        heapifyMin(a, n, smallest);
    }
}

void heapSortMin(int a[], int n){
    for (int i = n / 2 - 1; i >= 0; i--)
        heapifyMin(a, n, i);

    for (int i = n - 1; i >= 0; i--){
        int temp = a[0];
        a[0] = a[i];
        a[i] = temp;

        heapifyMin(a, i, 0);
    }
}

int main(){
    int a[] = {12, 3, 19, 6, 5};
    int n = sizeof(a) / sizeof(a[0]);
    heapSortMin(a, n);
    for (int i = 0; i < n; i++)
        printf("%d ", a[i]);
    return 0;
}
```

Output:



```
● PS C:\Users\sarva\OneDrive\Desktop\daa-lab> gcc minheapsort.c
● PS C:\Users\sarva\OneDrive\Desktop\daa-lab> ./a.exe
  19 12 6 5 3
○ PS C:\Users\sarva\OneDrive\Desktop\daa-lab>
```