Week 1: C++ basic information and algorithm analysis

4 October 2022

Course plan

https://ebs.sabis.sakarya.edu.tr/Ders/Detay/573162

1. Temel veri tipleri, veri kavramı ve algoritma analizi
2. Bellek yönetimi ve göstericiler
3. Özyineleme kavramı ve Özyinelemeli algoritmalar
4. Liste veri yapısı, Statik ve Dinamik diziler
5. Bağlı liste, tek yönlü bağlı listeler
6. Çift yönlü bağlı listeler, Dairesel listeler
7. Yığıt veri yapısı ve uygulamaları
8. Kuyruk veri yapısı, doğrusal kuyruk, dairesel kuyruk
9. Ağaç veri yapısı
10. İkili ağaç, İfade ağacı ve İkili arama ağacı
11. Öncelikli kuyruk ve Heap ağacı
12. AVL ağaçları
13. Genel Ağaç uygulamaları, Huffman, LempelZiv
14. Hash tabloları

Değerlendirme Sistemi

	Çalışma Tipi	Oran
☐	1. Ara Sınav	%20
☐	1. Kısa Sınav	%10
☐	1. Ödev	%40
☐	2. Ödev	%30
☐	1. Final	%40

Dr. İbrahim Delibaşoğlu
Department of Software Engineering
Sakarya University

References

Prof. Dr. Nejat YUMUŞAK, M. Fatih ADAK, "C/C++ ile Veri Yapıları ve Çözümlü Uygulamalar", Seçkin yayıncılık, 2014.

https://web.karabuk.edu.tr/hakankutucu/BLM227/VER%C4%B0%20YAPILARI%20v.6.3.pdf

https://www.youtube.com/watch?v=BBpAmxU_NQo&t=1361s

Dr. İbrahim Delibaşoğlu
Department of Software Engineering
Sakarya University

Lecture plan

- C++ basic informations
- Data types
- Compiling
  - makefile
- Performance analysis

Dr. İbrahim Delibaşoğlu
Department of Software Engineering
Sakarya University

C++ basic informations

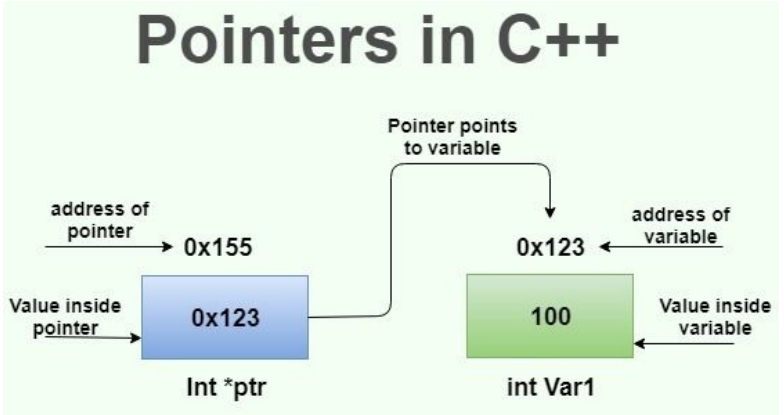
- C++ programming language will be used to implement and test the algorithm and data structures
- The details of the C++ programming is not the objective of the course
- The main upgradation from C to C++ is object-oriented programming.
- C++ is called a lower level language compared to languages like Java and C#.
- C doesn’t support OOP

Dr. İbrahim Delibaşoğlu
Department of Software Engineering
Sakarya University

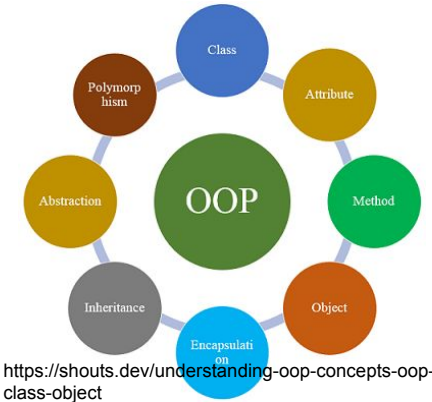
C++ basic informations

Pointers are powerful features of C++ that enables you to manipulate the data in the computer's memory directly.

Pointers are powerful features of C++ that enables you to manipulate the data in the computer's memory directly



https://simplesnippets.tech/cpp-pointers-concept-with-example/



https://shouts.dev/understanding-oop-concepts-oop-class-object

Dr. İbrahim Delibaşoğlu
Department of Software Engineering
Sakarya University

C++ basic informations

What is a compiler?

Computers understand only one language and that language consists of sets of instructions made of ones and zeros. This computer language is appropriately called ***machine language***.

Programming a computer directly in machine language using only ones and zeros is very tedious and error prone. To make programming easier, high level languages have been developed. High level programs also make it easier for programmers to inspect and understand each other's programs easier.

```
1 int a, b, sum;
2
3 cin >> a;
4 cin >> b;
5
6 sum = a + b;
7 cout << sum << endl;
```

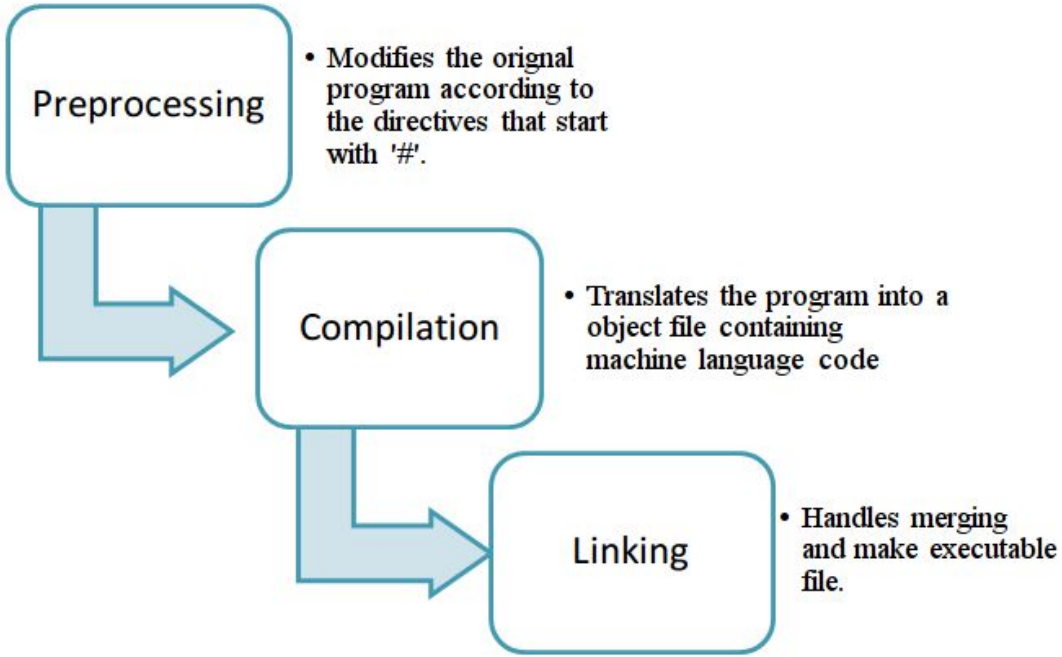
C++ is designed to be a compiled language, meaning that it is generally translated into machine language that can be understood directly by the system, making the generated program highly efficient.

https://cplusplus.com/doc/tutorial/introduction/

Dr. İbrahim Delibaşoğlu
Department of Software Engineering
Sakarya University

C++ basic informations

The C++ compiler (for example, MinGW) converts an executable C++ code into a form that the computer can run directly. This form is called machine language.



Dr. İbrahim Delibaşoğlu
Department of Software Engineering
Sakarya University

C++ basic informations

The best way to learn a programming language is by writing programs. Typically, the first program beginners write is a program called "Hello World", which simply prints "Hello World" to your computer screen. Although it is very simple, it contains all the fundamental components C++ programs have:

```
// my first program in C++
#include <iostream>

int main()
{
  std::cout << "Hello World!";
}
```

```
Hello World!
```

Dr. İbrahim Delibaşoğlu
Department of Software Engineering
Sakarya University

Compiling

```cpp
#include<iostream>

class Dikdortgen
{
public:
    Dikdortgen(int genislik,int yukseklik)
    {
        m_Genislik = genislik;
        m_Yukseklik= yukseklik;
    }
    Dikdortgen()
    {
        m_Genislik = 10;
        m_Yukseklik =20;
    }
    int genislik()
    {
        return m_Genislik;
    }
    int yukseklik()
    {
        return m_Yukseklik;
    }
private:
    int m_Genislik;
    int m_Yukseklik;
};

int main()
{
    Dikdortgen d1(40,60);

    using namespace std;

    cout<<"d1.genislik :"<<d1.genislik()<<endl;
    cout<<"d1.yukseklik:"<<d1.yukseklik()<<endl;

    cout<<"-----------------------------------"<<endl;
    Dikdortgen d2;
```

g++ command is a GNU c++ compiler invocation command, which is used for preprocessing, compilation, assembly and linking of source code to generate an executable file.

```
ibrahim@ibrahim: ~/Desktop/Dersler/Veri Yapıları/
File  Edit  View  Search  Terminal  Help
ibrahim@ibrahim:~/Desktop/Dersler/Veri Yapıları/1.Hafta/Kod1$ g++ main.cpp -o main
ibrahim@ibrahim:~/Desktop/Dersler/Veri Yapıları/1.Hafta/Kod1$ ./main
d1.genislik :40
d1.yukseklik:60
-----------------------------------
d2.genislik :10
d2.yukseklik:20
ibrahim@ibrahim:~/Desktop/Dersler/Veri Yapıları/1.Hafta/Kod1$
```

Dr. İbrahim Delibaşoğlu
Department of Software Engineering
Sakarya University

Compiling

```cpp
#ifndef Dikdortgen_hpp
#define Dikdortgen_hpp

class Dikdortgen
{
public:
    Dikdortgen(int genislik,int yukseklik);

    Dikdortgen();

    int genislik();

    int yukseklik();

private:
    int m_Genislik;
    int m_Yukseklik;
};



#endif
```

```cpp
#include "Dikdortgen.hpp"
Dikdortgen::Dikdortgen(int genislik,int yukseklik)
{
    m_Genislik = genislik;
    m_Yukseklik= yukseklik;
}
Dikdortgen::Dikdortgen()
{
    m_Genislik = 10;
    m_Yukseklik =20;
}
int Dikdortgen::genislik()
{
    return m_Genislik;
}
int Dikdortgen::yukseklik()
{
    return m_Yukseklik;
}
```

Dr. İbrahim Delibaşoğlu
Department of Software Engineering
Sakarya University

Compiling

```cpp
#include<iostream>
#include "Dikdortgen.hpp"

int main()
{
    Dikdortgen d1(40,60);

    using namespace std;

    cout<<"d1.genislik :"<<d1.genislik()<<endl;
    cout<<"d1.yukseklik:"<<d1.yukseklik()<<endl;

    cout<<"-----------------------------------"<<endl;
    Dikdortgen d2;

    cout<<"d2.genislik :"<<d2.genislik()<<endl;
    cout<<"d2.yukseklik:"<<d2.yukseklik()<<endl;
}
```

```
ibrahim@ibrahim:~/Desktop/Dersler/Veri Yapıları/1.Hafta/Kod2$
g++ main.cpp Dikdortgen.cpp -o main
ibrahim@ibrahim:~/Desktop/Dersler/Veri Yapıları/1.Hafta/Kod2$
./main
d1.genislik :40
d1.yukseklik:60
-----------------------------------
d2.genislik :10
d2.yukseklik:20
ibrahim@ibrahim:~/Desktop/Dersler/Veri Yapıları/1.Hafta/Kod2$
```

Dr. İbrahim Delibaşoğlu
Department of Software Engineering
Sakarya University

Compiling

A *makefile* is a text file that contains instructions for how to compile and link (or *build*) a set of source code files. A program (often called a *make* program) reads the makefile and invokes a compiler, linker, and possibly other programs to make an executable file. The Microsoft program is called NMAKE.

makefile

home > ibrahim > Desktop > Dersler > Veri Yapıları > 1.Hafta > K

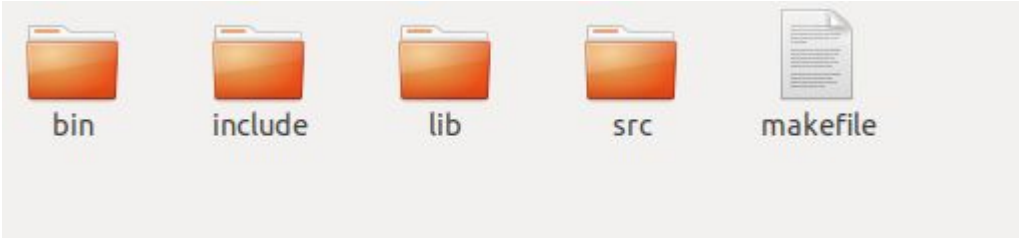
```
1  all:
2      g++ -c Dikdortgen.cpp
3      g++ -c main.cpp
4      g++ Dikdortgen.o main.o -o program
```

> ibrahim > Desktop > Dersler > Veri Yapıları > 1.Hafta > Koc

```
all:derle bagla calistir
derle:
    g++ -c Dikdortgen.cpp
    g++ -c main.cpp
bagla:
    g++ Dikdortgen.o main.o -o program
calistir:
    ./program
```

Dr. İbrahim Delibaşoğlu
Department of Software Engineering
Sakarya University

Compiling




```
all:derle bagla calistir
derle:
    g++ -c -I "./include" ./src/Dikdortgen.cpp -o ./lib/Dikdortgen.o
    g++ -c -I "./include" ./src/main.cpp -o ./lib/main.o
bagla:
    g++ ./lib/Dikdortgen.o ./lib/main.o -o ./bin/program
calistir:
    ./bin/program
```

n Delibaşoğlu
Department of Software Engineering
Sakarya University

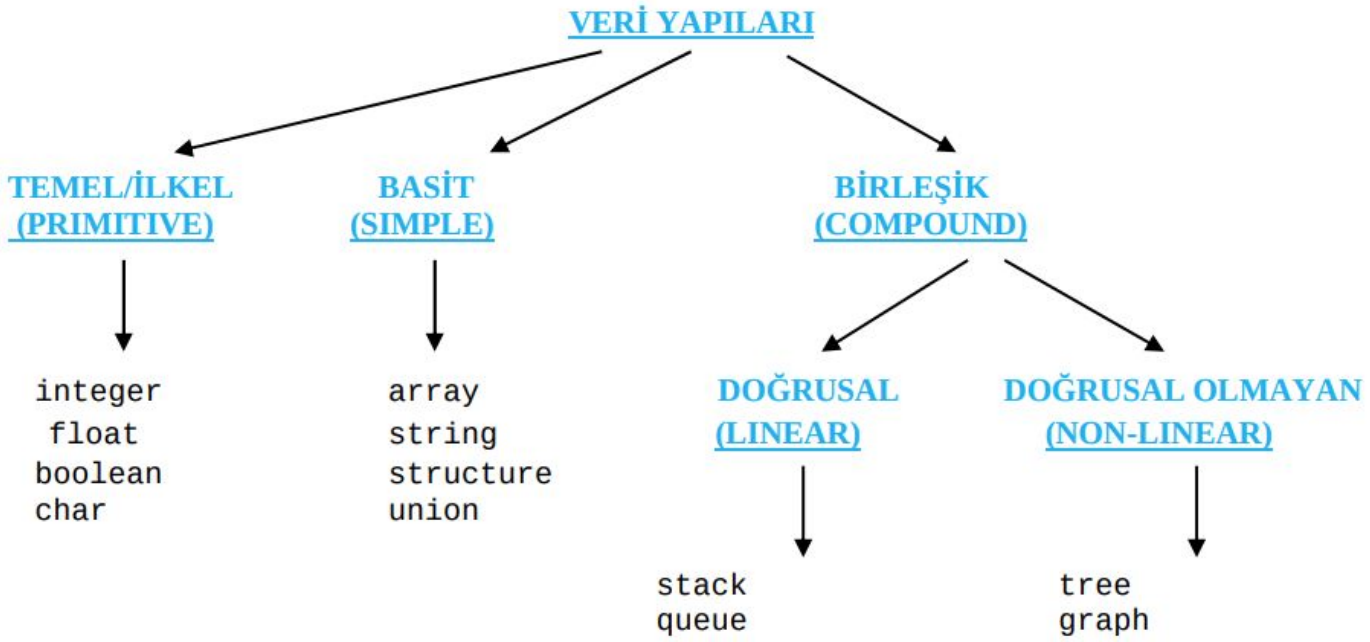
Data types

Tipi	Bit Boyutu	Tanım Aralığı
char	8	-127 – 127
unsigned char	8	0 – 255
signed char	8	-127 – 127
int	16 veya 32*	-32,767 – 32,767
unsigned int	16 veya 32*	0 – 65,535
signed int	16 veya 32*	-32,767 – 32,767
short int	16	-32,767 – 32,767
unsigned shortint	16	0 – 65,535
signed short int	16	-32,767 – 32,767
long int	32	-2,147,483,647 - 2,147,483,647
signed long int	32	-2,147,483,647 - 2,147,483,647
unsigned long int	32	0 – 4,294,967,295
float	32	3.4×10^{-38} - 3.4×10^{38}
double	64	1.7×10^{-308} – 1.7×10^{308}

Data types and ranges

Dr. İbrahim Delibaşoğlu
Department of Software Engineering
Sakarya University

Data types



Data types in C

Dr. İbrahim Delibaşoğlu
Department of Software Engineering
Sakarya University

Storing data

Data is stored in computer memory as a string of "Bits" consisting of 1s and 0s. The meaning of data in bit string form is revealed from the structure of the data.

If we consider the following 32-bit data; if this data is converted to ASCII data structure, each 8-bit data group corresponds to one character;

0100 0010	0100 0001	0100 0010	0100 0001
B	A	B	A

If this data is an unsigned 16-bit integer, each 16-bit data corresponds to an unsigned integer;

0100 0010 0100 0001	0100 0010 0100 0001
16961	16961

Dr. İbrahim Delibaşoğlu
Department of Software Engineering
Sakarya University

Performance analysis

Algorithm complexity (Big-O) is used to describe the performance of an algorithm.

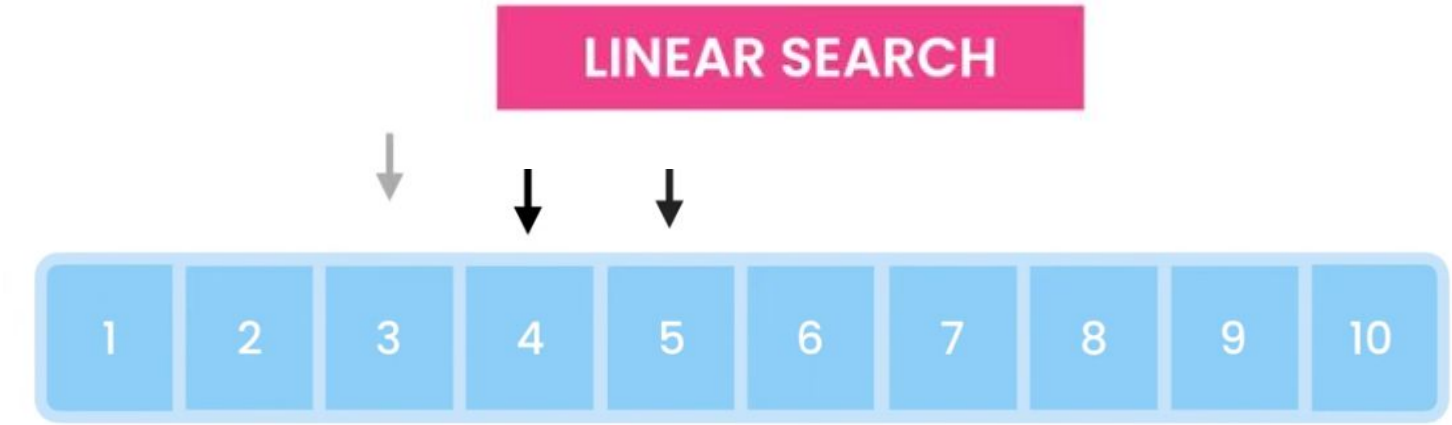
O(n) and data structures relation?

Dr. İbrahim Delibaşoğlu
Department of Software Engineering
Sakarya University

Performance analysis

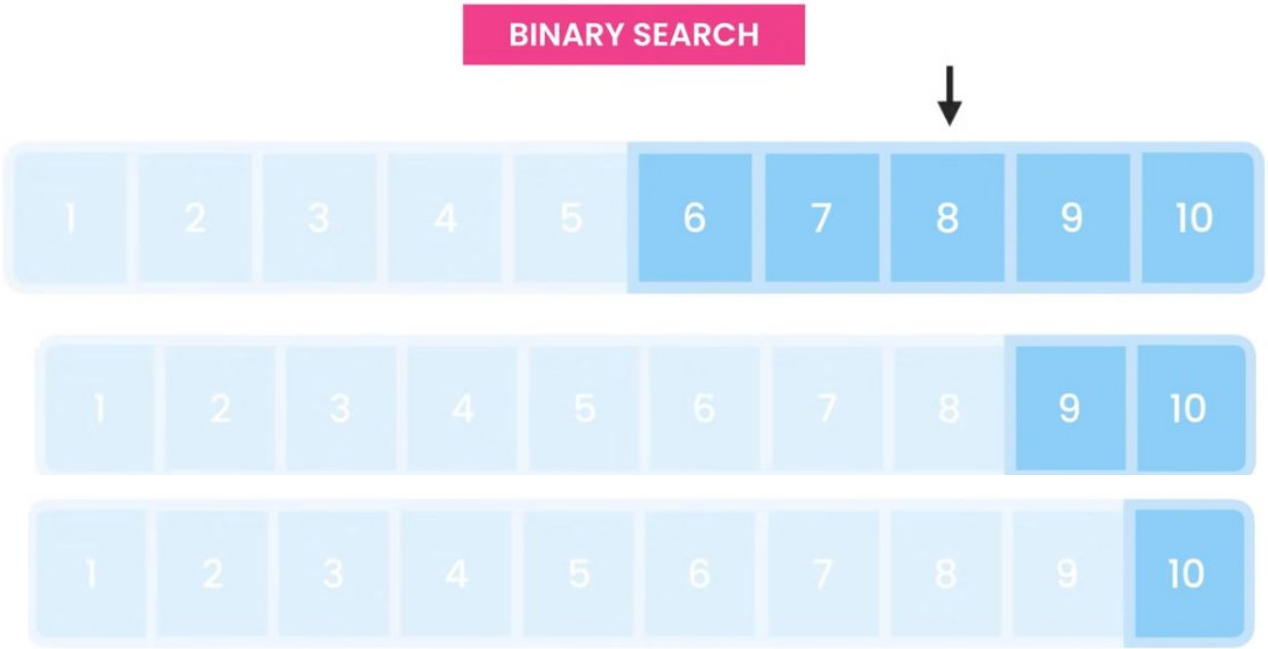
Dr. İbrahim Delibaşoğlu
Department of Software Engineering
Sakarya University

Performance analysis



Dr. İbrahim Delibaşoğlu
Department of Software Engineering
Sakarya University

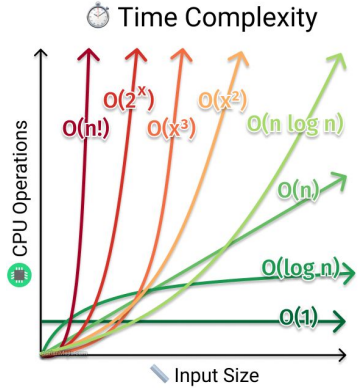
Performance analysis



Dr. İbrahim Delibaşoğlu
Department of Software Engineering
Sakarya University

Performance analysis

- Algorithmic complexity is a measure of how long an algorithm would take to complete given an input of size n.
- If an algorithm has to scale, it should compute the result within a finite and practical time bound even for large values of n.
- For this reason, complexity is calculated asymptotically as n approaches infinity.
- While complexity is usually in terms of time, sometimes complexity is also analyzed in terms of space, which translates to the algorithm's memory requirements
- Analysis of an algorithm's complexity is helpful when comparing algorithms or seeking improvements.



https://adrianmejia.com/how-to-find-time-complexity-of-an-algorithm-code-big-o-notation/

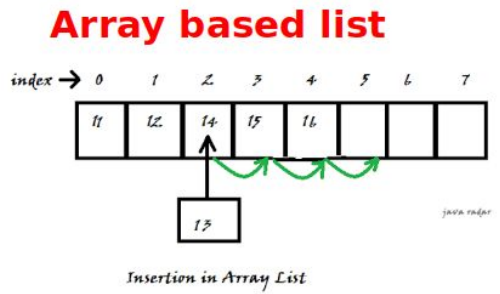
Algorithm	Time Complexity			Space Complexity
	Best	Average	Worst	Worst
Quicksort	Ω(n log(n))	θ(n log(n))	O(n^2)	O(log(n))
Mergesort	Ω(n log(n))	θ(n log(n))	O(n log(n))	O(n)
Timsort	Ω(n)	θ(n log(n))	O(n log(n))	O(n)
Heapsort	Ω(n log(n))	θ(n log(n))	O(n log(n))	O(1)
Bubble Sort	Ω(n)	θ(n^2)	O(n^2)	O(1)
Insertion Sort	Ω(n)	θ(n^2)	O(n^2)	O(1)
Selection Sort	Ω(n^2)	θ(n^2)	O(n^2)	O(1)
Tree Sort	Ω(n log(n))	θ(n log(n))	O(n^2)	O(n)
Shell Sort	Ω(n log(n))	θ(n(log(n))^2)	O(n(log(n))^2)	O(1)
Bucket Sort	Ω(n+k)	θ(n+k)	O(n^2)	O(n)
Radix Sort	Ω(nk)	θ(nk)	O(nk)	O(n+k)
Counting Sort	Ω(n+k)	θ(n+k)	O(n+k)	O(k)
Cubesort	Ω(n)	θ(n log(n))	O(n log(n))	O(n)

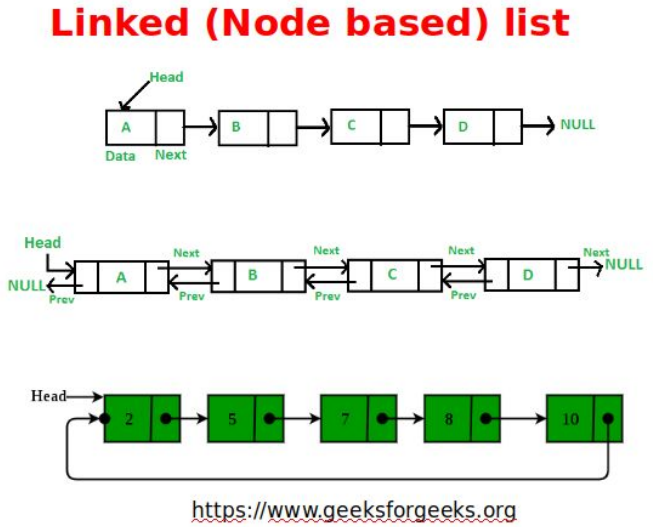
https://devopedia.org/algorithmic-complexity

Dr. İbrahim Delibaşoğlu
Department of Software Engineering
Sakarya University

Overview:Lists

- In computer science, a list or sequence is an abstract data type that represents a finite number of ordered values, where the same value may occur more than once.
- Many programming languages provide support for list data types, and have special syntax and semantics for lists and list operations
- Array based implementation
- Linked (Node based) implementation

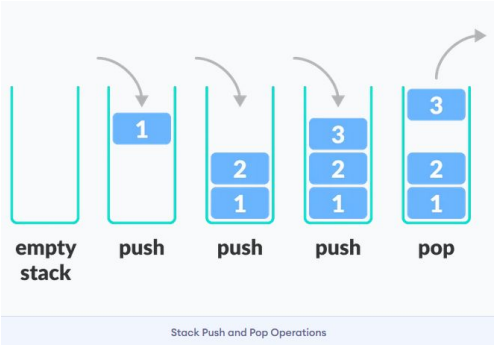




Overview:Stacks

- A stack is a linear data structure that follows the principle of Last In First Out (LIFO).
- This means the last element inserted inside the stack is removed first.
- You can think of the stack data structure as the pile of plates on top of another.

Stack representation similar to a pile of plate



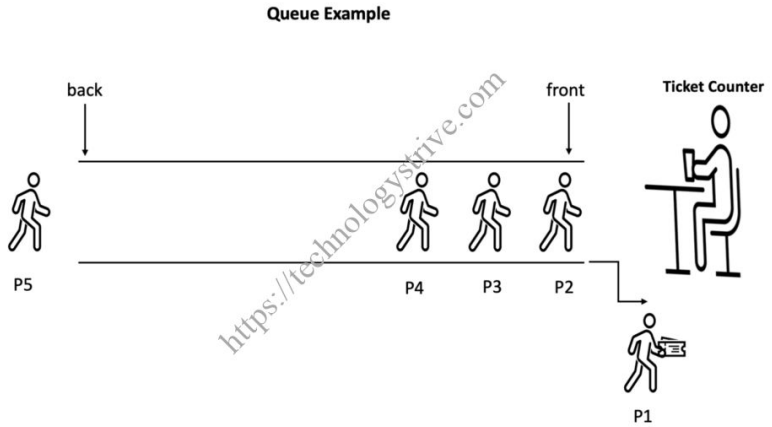
Stack Push and Pop Operations

https://www.programiz.com/dsa/stack

Dr. İbrahim Delibaşoğlu
Department of Software Engineering
Sakarya University

Overview:Queues

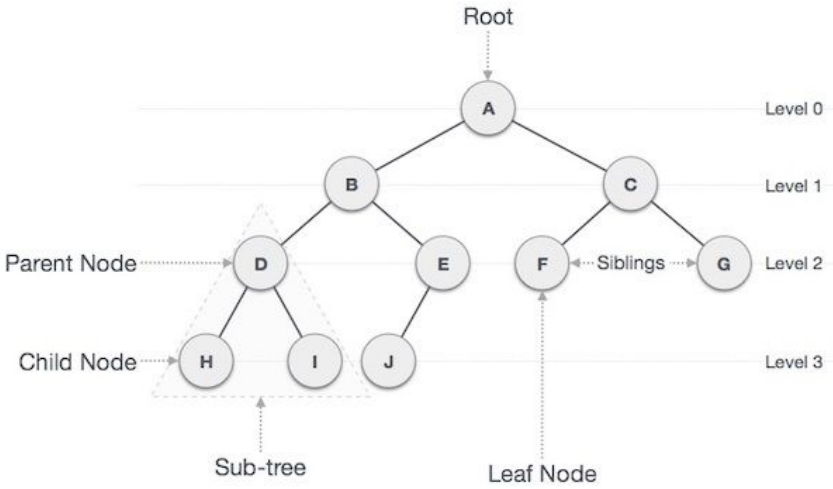
- Queue is an abstract data structure, somewhat similar to Stacks.
- Unlike stacks, a queue is open at both its ends.
- One end is always used to insert data (enqueue) and the other is used to remove data (dequeue).
- Queue follows First-In-First-Out methodology, i.e., the data item stored first will be accessed first.



Dr. İbrahim Delibaşoğlu
Department of Software Engineering
Sakarya University

Overview:Trees

- In computer science, a tree is a widely used abstract data type that simulates a hierarchical tree structure, with a root value and subtrees of children with a parent node, represented as a set of linked nodes.
- We will discuss binary tree or binary search tree specifically.
- Binary Tree is a special data structure used for data storage purposes.
- A binary tree has a special condition that each node can have a maximum of two children.



Dr. İbrahim Delibaşoğlu
Department of Software Engineering
Sakarya University

Thanks for your attention…

Dr. İbrahim Delibaşoğlu
Department of Software Engineering
Sakarya University