



Index Mapping (or Trivial Hashing) with negatives allowed

Last Updated : 02 Jun, 2023

Index Mapping (also known as Trivial Hashing) is a simple form of hashing where the data is directly mapped to an index in a hash table. The hash function used in this method is typically the identity function, which maps the input data to itself. In this case, the key of the data is used as the index in the hash table, and the value is stored at that index.

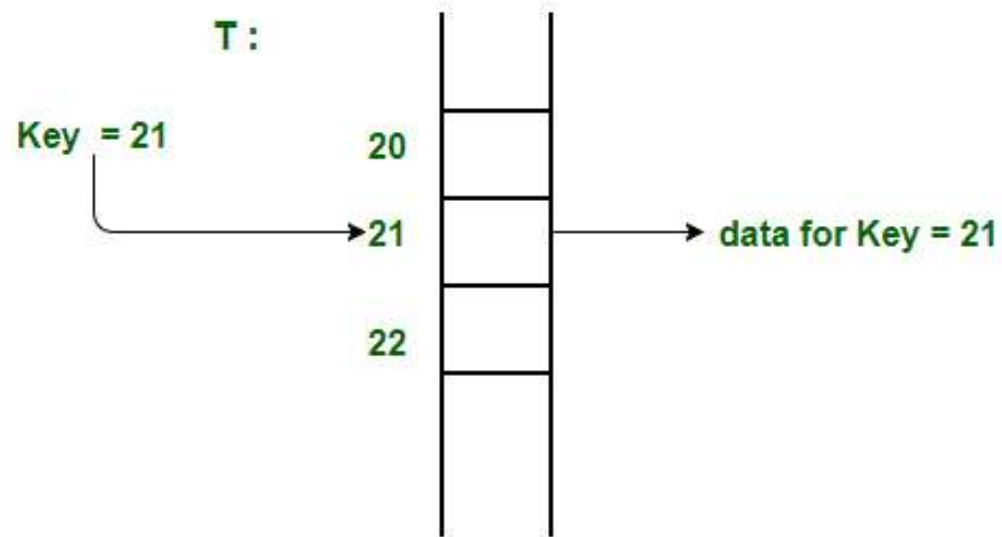
For example, if we have a hash table of size 10 and we want to store the value “apple” with the key “a”, the trivial hashing function would simply map the key “a” to the index “a” in the hash table, and store the value “apple” at that index.

One of the main advantages of Index Mapping is its simplicity. The hash function is easy to understand and implement, and the data can be easily retrieved using the key. However, it also has some limitations. The main disadvantage is that it can only be used for small data sets, as the size of the hash table has to be the same as the number of keys. Additionally, it doesn't handle collisions, so if two keys map to the same index, one of the data will be overwritten.

Given a limited range array contains both positive and non-positive numbers, i.e., elements are in the range from -MAX to +MAX. Our task is to search if some number is present in the array or not in $O(1)$ time.

Since the range is limited, we can use [index mapping](#) (or trivial hashing). We use values as the index

in a big array. Therefore we can search and insert elements in $O(1)$ time.



How to handle negative numbers?

The idea is to use a 2D array of size `hash[MAX+1][2]`

Algorithm:

Assign all the values of the hash matrix as 0.

Traverse the given array:


- *If the element `ele` is non negative assign*
 - *`hash[ele][0]` as 1.*
- *Else take the absolute value of `ele` and*
 - *assign `hash[ele][1]` as 1.*

To search any element `x` in the array.

- If `X` is non-negative check if `hash[X][0]` is 1 or not. If `hash[X][0]` is one then the number is present else not present.
- If `X` is negative take the absolute value of `X` and then check if `hash[X][1]` is 1 or not. If `hash[X][1]` is one then the number is present

Below is the implementation of the above idea.

C++

```
 // CPP program to implement direct index mapping  
// with negative values allowed.
```

```
#include <bits/stdc++.h>
using namespace std;
#define MAX 1000

// Since array is global, it is initialized as 0.
bool has[MAX + 1][2];

// searching if X is Present in the given array
// or not.
bool search(int X)
{
    if (X >= 0) {
        if (has[X][0] == 1)
            return true;
        else
            return false;
    }

    // if X is negative take the absolute
    // value of X.
    X = abs(X);
    if (has[X][1] == 1)
        return true;

    return false;
}

void insert(int a[], int n)
{
    for (int i = 0; i < n; i++) {
        if (a[i] >= 0)
            has[a[i]][0] = 1;
        else
            has[abs(a[i])][1] = 1;
    }
}
```

```
// Driver code
int main()
{
    int a[] = { -1, 9, -5, -8, -5, -2 };
    int n = sizeof(a)/sizeof(a[0]);
    insert(a, n);
    int X = -5;
    if (search(X) == true)
        cout << "Present";
    else
        cout << "Not Present";
    return 0;
}
```

Java

```
// Java program to implement direct index
// mapping with negative values allowed.
class GFG
{
    final static int MAX = 1000;

    // Since array is global, it
    // is initialized as 0.
    static boolean[][] has = new boolean[MAX + 1][2];

    // searching if X is Present in
    // the given array or not.
    static boolean search(int X)
    {
        if (X >= 0)
        {
```

```
        if (has[X][0] == true)
        {
            return true;
        }
        else
        {
            return false;
        }
    }

    // if X is negative take the
    // absolute value of X.
    X = Math.abs(X);
    if (has[X][1] == true)
    {
        return true;
    }

    return false;
}

static void insert(int a[], int n)
{
    for (int i = 0; i < n; i++)
    {
        if (a[i] >= 0)
        {
            has[a[i]][0] = true;
        }
        else
        {
            int abs_i = Math.Abs(a[i]);
            has[abs_i][1] = true;
        }
    }
}
```

```
// Driver code
public static void main(String args[])
{
    int a[] = {-1, 9, -5, -8, -5, -2};
    int n = a.length;
    insert(a, n);
    int X = -5;
    if (search(X) == true)
    {
        System.out.println("Present");
    }
    else
    {
        System.out.println("Not Present");
    }
}

// This code is contributed
// by 29AjayKumar
```

Python3

```
# Python3 program to implement direct index
# mapping with negative values allowed.

# Searching if X is Present in the
# given array or not.
def search(X):

    if X >= 0:
        return has[X][0] == 1
```

```
# if X is negative take the absolute
# value of X.
X = abs(X)
return has[X][1] == 1

def insert(a, n):

    for i in range(0, n):
        if a[i] >= 0:
            has[a[i]][0] = 1
        else:
            has[abs(a[i))][1] = 1

# Driver code
if __name__ == "__main__":

    a = [-1, 9, -5, -8, -5, -2]
    n = len(a)

    MAX = 1000

    # Since array is global, it is
    # initialized as 0.
    has = [[0 for i in range(2)]
            for j in range(MAX + 1)]
    insert(a, n)

    X = -5
    if search(X) == True:
        print("Present")
    else:
        print("Not Present")

# This code is contributed by Rituraj Jain
```


C#



Javascript



```
// JavaScript program to implement direct index  
// mapping with negative values allowed.
```

```
let MAX = 1000;
```

```
// Since array is global, it  
// is initialized as 0.
```

```
let has = new Array(MAX+1);
```

```
for(let i=0;i<MAX+1;i++)
```

```
{
```

```
    has[i]=new Array(2);
```

```
    for(let j=0;j<2;j++)
```

```
        has[i][j]=0;
```

```
}
```

```
// searching if X is Present in
```

```
// the given array or not.
```

```
function search(X)
```

```
{
```

```
    if (X >= 0)
```

```
{
```

```
        if (has[X][0] == true)
        {
            return true;
        }
        else
        {
            return false;
        }
    }

    // if X is negative take the
    // absolute value of X.
    X = Math.abs(X);
    if (has[X][1] == true)
    {
        return true;
    }

    return false;
}

function insert(a,n)
{
    for (let i = 0; i < n; i++)
    {
        if (a[i] >= 0)
        {
            has[a[i]][0] = true;
        }
        else
        {
            let abs_i = Math.abs(a[i]);
            has[abs_i][1] = true;
        }
    }
}
```

```
// Driver code
let a=[-1, 9, -5, -8, -5, -2];
let n = a.length;
    insert(a, n);
    let X = -5;
    if (search(X) == true)
    {
        document.write("Present");
    }
    else
    {
        document.write("Not Present");
    }

// This code is contributed by rag2127
// corrected by akashish__
```

Output

Present

Time Complexity: The time complexity of the above algorithm is $O(N)$, where N is the size of the given array.

Space Complexity: The space complexity of the above algorithm is $O(N)$, because we are using an array of max size.

"The DSA course helped me a lot in clearing the interview rounds. It was really very helpful in setting a strong foundation for my problem-solving skills. Really a great investment, the passion Sandeep sir has towards DSA/teaching is what made the huge difference." - **Gaurav | Placed at Amazon**

Before you move on to the world of development, **master the fundamentals of DSA** on which every advanced algorithm is built upon. Choose your preferred language and start learning today:

[DSA In JAVA/C++](#)

[DSA In Python](#)

[DSA In JavaScript](#)

Trusted by Millions, Taught by One- Join the best DSA Course Today!

Recommended Problems

Frequently asked DSA Problems

202

Previous

What is Hashing?

Solve Problems

[Suggest improvement](#)

Next

**Separate Chaining Collision Handling
Technique in Hashing**

Share your thoughts in the comments

Add Your Comment

Similar Reads

Sorting using trivial hash function

Sum of Fibonacci Numbers with alternate negatives

Maximize the difference between two subsets of a set with negatives

Remove all negatives from the given Array

Minimum index i such that all the elements from index i to given index are equal

Minimum cost to reach end of array when a maximum jump of K index is allowed

Find a Fixed Point (Value equal to index) in a given array | Duplicates Allowed

Efficient method to store a Lower Triangular Matrix using row-major mapping

Efficient method to store a Lower Triangular Matrix using Column-major mapping

Check if mapping is possible to make sum of first array larger than second array



GeeksforGeeks

Article Tags : [Arrays](#), [DSA](#), [Hash](#)

Practice Tags : [Arrays](#), [Hash](#)



Company

[About Us](#)

Explore

Languages

[Python](#)

DSA

[Data Structures](#)

**Data Science &
ML**

**Web
Technologies**

Legal	Job-A-Thon Hiring	Java	Algorithms	Data Science With	HTML
Careers	Challenge	C++	DSA for Beginners	Python	CSS
In Media	Hack-A-Thon	PHP	Basic DSA Problems	Data Science For	JavaScript
Contact Us	GfG Weekly Contest	GoLang	DSA Roadmap	Beginner	TypeScript
Advertise with us	Offline Classes	SQL	DSA Interview	Machine Learning	ReactJS
GFG Corporate	(Delhi/NCR)	R Language	Questions	Tutorial	NextJS
Solution	DSA in JAVA/C++	Android Tutorial	Competitive	ML Maths	NodeJs
Placement Training	Master System		Programming	Data Visualisation	Bootstrap
Program	Design			Tutorial	Tailwind CSS
	Master CP			Pandas Tutorial	
	GeeksforGeeks			NumPy Tutorial	
	Videos			NLP Tutorial	
	Geeks Community			Deep Learning	
				Tutorial	

Python Tutorial

Python Programming
Examples

Django Tutorial

Python Projects

Python Tkinter

Web Scraping

OpenCV Tutorial

Python Interview
Question

**Computer
Science**

GATE CS Notes

Operating Systems

Computer Network

Database

Management System

Software Engineering

Digital Logic Design

Engineering Maths

DevOps

Git

AWS

Docker

Kubernetes

Azure

GCP

DevOps Roadmap

System Design

High Level Design

Low Level Design

UML Diagrams

Interview Guide

Design Patterns

OOAD

System Design

Bootcamp

Interview Questions

School Subjects

Mathematics

Physics

Chemistry

Biology

Social Science

English Grammar

Commerce

Accountancy

Business Studies

Economics

Management

HR Management

Finance

Income Tax

**UPSC Study
Material**

Polity Notes
Geography Notes
History Notes
Science and
Technology Notes
Economy Notes
Ethics Notes
Previous Year Papers

**Preparation
Corner**

Company-Wise
Recruitment Process
Resume Templates
Aptitude Preparation
Puzzles
Company-Wise
Preparation
Companies
Colleges

**Competitive
Exams**

JEE Advanced
UGC NET
SSC CGL
SBI PO
SBI Clerk
IBPS PO
IBPS Clerk

More Tutorials

Software
Development
Software Testing
Product Management
Project Management
Linux
Excel
All Cheat Sheets

Free Online Tools

Typing Test
Image Editor
Code Formatters
Code Converters
Currency Converter
Random Number
Generator
Random Password
Generator

Write & Earn

Write an Article
Improve an Article
Pick Topics to Write
Share your
Experiences
Internships

@GeeksforGeeks, Sanchhaya Education Private Limited, All rights reserved