# Pair in C++

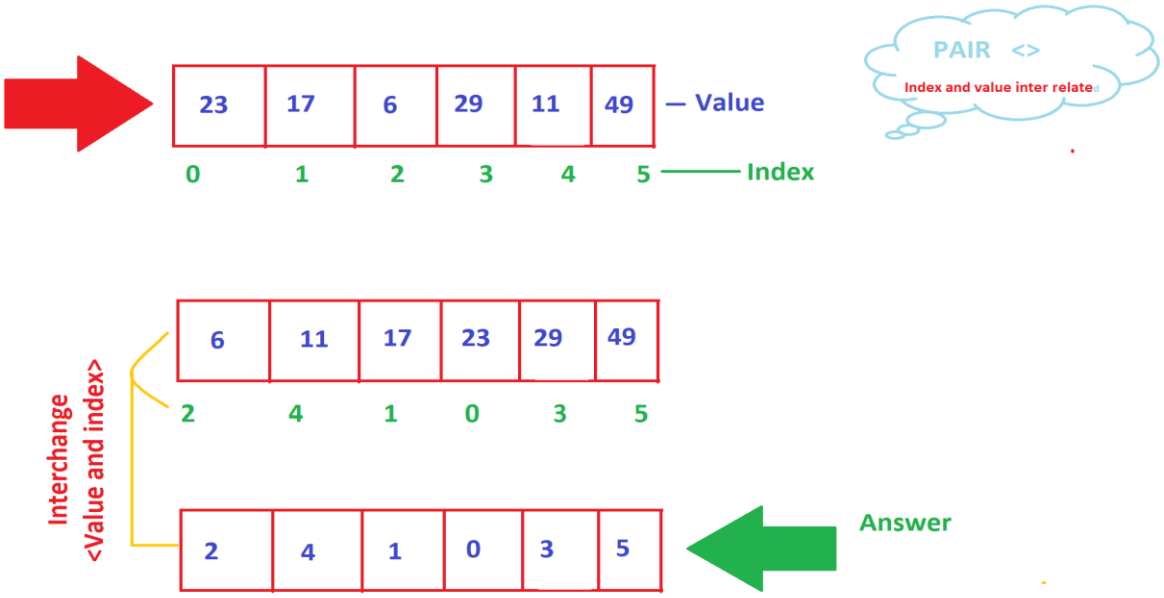
Jaydeep Shah (radhey04ec@gmail.com)

In the realm of C++, the concept of "pair" proves to be useful when there is a **relationship between two values that may belong to different data types**.
To illustrate, imagine having two buckets, one filled with fruits and vegetables, and the other containing MRP prices per kilogram.
In order to establish a connection between these buckets for future utilization, the "pair" concept in C++ comes into play.

## Code

```
/*
Application of Pair in C++
Jaydeep Shah
*/

#include<iostream>
using namespace std;


int main()
{
	//SYNTAX:
	//pair <data_type1, data_type2> Pair_name (value1, value2) ;


	//-------------------------------------------------------------
	//Create pair of same type
	pair<int,int> p;

	//Assign value of first and second element
	p.first	= 10;
	p.second		= 20;
	//print out that value
	cout<< "Pair value of p = "<<p.first<<" and "<<p.second<<endl;
	//-------------------------------------------------------------

	//-------------------------------------------------------------
	//Create pair of different type
	pair<char,int> p1;
	p1.first = 'A';
	p1.second = 1;
	cout<< "Pair value p1 = "<<p1.first<<" and "<<p1.second<<endl;
	//-------------------------------------------------------------

	//-------------------------------------------------------------
	//Assign value during declaration time
	pair<float,string> p3(3.14,"Pi");
	cout<< "Pair value p3 = "<<p3.first<<" and "<<p3.second<<endl;
	//-------------------------------------------------------------


	return 0;
}
```

Sort and Interchange




**Application**

**Code:**

```
//Application of pair and vector
//Sorting Array as per value and then change the value with related indexes.
//Here array and it's value both are inter related,so during sorting as per value need to store related index also.
//Here concept of pair and vector is very useful.
```

<bits/stdc++.h> is header file, that includes all other standard library header file, this is useful to save time.

But note that,it increases compilation time. And also increase difficulty to maintain and optmize code, because of we may not aware all functions included by this library.

```
#include<iostream>
#include <bits/stdc++.h>     //Required for sorting
#include<vector>

using namespace std;

//------------------------------------------------------------------------------
//Our own function
bool compare_array_val(pair<int,int>p1,pair<int,int>p2)
{
        return(p1.first < p2.first);
}
//------------------------------------------------------------------------------
```

```
int main()
{
	//------------------------------------------------------------------------
	//Fixed Array
	int array[] = {23,17,6,29,11,49};
	vector< pair<int,int> > v;	//Like 2D array but here every elemnt have significant value with it's pair element

	//Store value and index as a pair of vector
	for(int a = 0;a < (sizeof(array) / sizeof(array[0])); a++)
	{
		//make pair is inbuilt function
		v.push_back(make_pair(array[a],a));
			//We are storing first value and second index
	}
	cout << endl << "Before sorting : " << endl;
	vector< pair<int , int> > :: iterator it;
	for(it=v.begin();it != v.end(); it++)
	{
		cout << endl << "Element : " << it->first << " Index : " << it->second;
	}

	//------------------------------------------------------------------------


	//------------------------------------------------------------------------
	//Sort as per value
	sort(v.begin(),v.end(),compare_array_val);
	//------------------------------------------------------------------------


	//------------------------------------------------------------------------
	//print array
	//After sorting element and related index
	vector < pair<int,int> > :: iterator itr;
	cout<<endl<<"After sorting : "<<endl;
	for(itr = v.begin();itr != v.end(); itr++)
	{
		cout<< endl << "Element : " << itr->first << " at index : " << itr->second;
	}
	//------------------------------------------------------------------------

	//------------------------------------------------------------------------
	//Interchange with index
	for(itr = v.begin();itr != v.end(); itr++)
	{
		int temp = itr->first;
		itr->first = itr->second;
		itr->second = temp;
	}
	//------------------------------------------------------------------------
```

```
	//---------------------------------------------------------------------------
	cout << endl << "After interchange :"<<endl;
	for(itr = v.begin();itr != v.end(); itr++)
	{
		cout<< endl << "Element : " << itr->first << " at index : " << itr->second;
	}
	//---------------------------------------------------------------------------

	return 0;
}
```

```
Before sorting :

Element : 23 Index : 0
Element : 17 Index : 1
Element : 6 Index : 2
Element : 29 Index : 3
Element : 11 Index : 4
Element : 49 Index : 5
After sorting :

Element : 6 at index : 2
Element : 11 at index : 4
Element : 17 at index : 1
Element : 23 at index : 0
Element : 29 at index : 3
Element : 49 at index : 5
After interchange :

Element : 2 at index : 6
Element : 4 at index : 11
Element : 1 at index : 17
Element : 0 at index : 23
Element : 3 at index : 29
Element : 5 at index : 49
--------------------------------
```