PAIRS

PAIRS :-> Stores pair of values of same/different data types. And The pair container is a simple container defined in **<utility>** header file.

How to declare?

Syntax : pair< data_type1, data_type2 > pair_name;

E.g. pair <int , string> myPair;

How to initialize pair?

Method(1): using make_pair() function:

```
pair<int, string> myPair;
myPair = make_pair(1,"himanshu");
```

Method(2): using Braces:

```
pair<int, string> myPair = {1, "himanshu"};
```

Method(3): constructor notation:

```
pair<int, string> myPair(1,"himanshu");
```

What if don't initialize our pairs?

Example.

```
pair<int, double> myPair;
cout<<myPair.first<< "  "<<myPair.second;

OUTPUT: 0 0
```

COPYING ONE PAIR INTO ANOTHER PAIR:

Shallow copy**:** it copies references, so if we make changes in one of object then they will also reflect in another object.

Deep copy: it copies only values, so if we make changes in any one of object then those changes will not reflect in another object.

So here, when we make one pair equal to another. So its a DEEP COPY .

For e.g.

```
pair<int, int> pair1 = {4,5};
pair<int, int> pair2;
// COPYING pair1 into pair2:
pair2 = pair1;
// printing pair2
cout<<"initially:";
cout<<"pair1 = "<<pair1.first<<" "<<pair1.second<<endl;
cout<<"pair2 = "<<pair2.first<<" "<<pair2.second<<endl;
// lets make some change in pair2 and see will they reflect in pair1
pair2.second = 96;
cout<<"Printing after making some changes: "<<endl;
cout<<"pair1 = "<<pair1.first<<" "<<pair1.second<<endl;
cout<<"pair2 = "<<pair2.first<<" "<<pair2.second<<endl;
```

Output:

```
intially:pair1 = 4 5
pair2 = 4 5
Printing after making some changes:
pair1 = 4 5
pair2 = 4 96
```

Hence we conclude that, This is an example of deep copy(i.e. making changes in one object will not reflect in another object).

ARRAY OF PAIRS:

```
// array of pairs:

pair<int, int> pair_arr[3];
// initializing the array of pairs:
for (int i = 0; i < 3; i++)
{
    cin >> pair_arr[i].first >> pair_arr[i].second;
}
cout<<"\n Printing values using for-each loop: "<<endl;
for (pair<int, int> item : pair_arr)
{
    cout << item.first << " " << item.second << endl;
}
```

OUTPUT:

```
7 5 3 6 9 4

Printing values using for-each loop:
7 5
3 6
9 4
```

Here we also noticed that how we can use ‘for-each’ loop:

```
for (pair<int, int> item : pair_arr)
{
    cout << item.first << " " << item.second << endl;
}
```

How to access both the elements of pairs?
Ans:
-> for accessing first element of pair: use .first
-> for accessing second element : use .second

Member function related to pairs:

(1) make_pair() : used to initialize pairs

(2) swap() : swaps the corresponding elements of two pairs of same type.

(3) sort() : sorts the pairs according to first element by default.
Example:

```
(1, B)
(2, A)
(3, F)
(4, D)
(5, C)
(6, G)
(7, H)
(8, E)
```

---

4. **operators(=, ==, !=, >=, <=) :** We can use operators with pairs as well :

Example:

```
pair<int, int> pair1(1, 12);
pair<int, int> pair2(9, 12);

cout << (pair1 == pair2) << endl;
cout << (pair1 != pair2) << endl;
// for logical operators it returns 0 or 1
// by only comparing the first value of the pair.
cout << (pair1 >= pair2) << endl;
cout << (pair1 <= pair2) << endl;
cout << (pair1 > pair2) << endl;
cout << (pair1 < pair2) << endl;


OUTPUT:
0
1
0
1
0
1
```

PROBLEMS

Problem: *Given two arrays a[] and b[] of equal size. The task is to sort the array b[] according to the elements of array a[]. That is, elements of the array b[] should be rearranged by following the corresponding elements of array a[] as appeared in sorted order.*

Input: a[] = {2, 1, 5, 4, 8, 3, 6, 7};
b[] = {'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H'}
Output: B A F D C G H E

Explanation:
Consider first elements of both arrays: (2, A)
Since the correct location of element 2 in a[] is at position 2.
Therefore, the corresponding element of b[] is also placed at position 2.
Similarly, the rest of the elements are arranged in the following way:
(1, B)
(2, A)
(3, F)
(4, D)
(5, C)
(6, G)
(7, H)
(8, E)

Code:

```cpp
#include <bits/stdc++.h>
using namespace std;

void pair_sort(int a[], char b[], int size)
{
    pair<int, int> myPair[size];
    // filing values inside array of pairs:
    for (int i = 0; i < size; i++)
    {
        myPair[i].first = a[i];
        myPair[i].second = b[i];
    }
    sort(myPair, (myPair + size));
    for (int i = 0; i < size; i++)
    {
        b[i] = myPair[i].second;
    }
}
int main()
{
    int a[] = {2, 1, 5, 4, 8, 3, 6, 7};
    char b[] = {'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H'};
    int size = sizeof(a) / sizeof(a[0]);
    pair_sort(a, b, size);
    for (int i = 0; i < size; i++)
    {
        cout << b[i] << "  ";
    }
}


OUTPUT:  B  A  F  D  C  G  H  E
```