Instructor created

6.19 Passing vector to a function

Extracted from GeeksforGeeks.com

When we pass an array to a function, a pointer is actually passed.

When a vector is passed to a function, a copy of the vector is created. For example, we can see the below program, changes made inside the function are not reflected outside because the function has a copy.

Code Editor 6.19.1: Passing 1D vector to a function by value

Click run.

Load default template...

```
// C++ program to demonstrate that when vectors
// are passed to functions without &, a copy is
// created.
#include <iostream>
#include <vector>
using namespace std;

// The vect here is a copy of vect in main()
void func(vector<int> vect)
{
   vect.push_back(30);
}

int main()
{
   vector<int> vect;
   vect.push_back(10);
   vect.push_back(20);
```

Run

The above style of passing might also take a lot of time in cases of large vectors. So it is a good idea to pass by reference.

Code Editor 6.19.2: Passing 1D vector to a function by reference

Click run.

Load default template...

```
// C++ program to demonstrate how vectors
// can be passed by reference.
#include <iostream>
#include <vector>
using namespace std;

// The vect is passed by reference and changes
// made here reflect in main()
void func(vector<int> &vect)
{
   vect.push_back(30);
}

int main()
{
   vector<int> vect;
   vect.push_back(10);
   vect.push_back(20);
```

Run

When a vector is passed by reference to a function, a copy of the vector is not created. For example, we can see below program, changes made inside the function are now reflected outside because function has a reference to the original vector.

If we do not want a function to modify a vector, we can pass it as a const reference.

Code Editor 6.19.3: Passing 1D vector to a function by reference

Click run.

Load default template...

```
// C++ program to demonstrate how vectors
// can be passed by reference with modifications
// restricted.
#include <iostream>
#include <vector>
using namespace std;

// The vect is passed by constant reference
// and cannot be changed by this function.
void func(const vector<int> &vect)
{
   vect.push_back(30); // Uncommenting this line would cause the
                       // below error

   for (int i=0; i<vect.size(); i++)
   cout << vect[i] << " ";
}
```

Run

Passing 2D vector to a function by value

Code Editor 6.19.4: Passing 2D vector to a function by value

Click run. [Example extracted from stackoverflow.com]

Load default template...

```
#include <iostream>
#include <vector>
using namespace std;

// print 2d vector
void printMatrix(vector < vector<int> > matrix)
{
    cout << "[";
    for(int i=0; i<matrix.size(); i++)
    {
        cout << "[" << matrix[i][0];
        for(int j=1; j<matrix[0].size(); j++)
        {
            cout  << ", " << matrix[i][j];
        }
        cout << "]" << endl;
    }
    cout << "]" << endl;
```

Run

Here is how to passing 2D vector to a function by reference.

Code Editor 6.19.5: Passing 2D vector to a function by reference

Click run.

Load default template...

```
#include <iostream>
#include <vector>
using namespace std;

// print 2d vector by reference
void printMatrix(vector < vector<int> >& matrix)
{
    cout << "[";
    for(int i=0; i<matrix.size(); i++)
    {
        cout << "[" << matrix[i][0];
        for(int j=1; j<matrix[0].size(); j++)
        {
            cout  << ", " << matrix[i][j];
        }
        cout << "]" << endl;
    }
    cout << "]" << endl;
```

Run