MAPS

```
#include<iostream>
#include<string>
#include<map>
using namespace std;

int main()
{
	map<string, char> grade_list;
	grade_list["satish"] = 'A';
	grade_list["Sam"] = 'B';
	map<string, char>::iterator map_it;
	map_it = grade_list.begin();

	cout << map_it->first;
	cout << map_it->second;
	map_it++;
	cout << map_it->first;
	cout << map_it->second;
	system("pause");
	return 0;
}
```

Updating an existing record

```
#include<iostream>
#include<string>
#include<map>
using namespace std;

int main()
{
	map<string, char> grade_list;
	grade_list["satishcj"] = 'A';
	grade_list["Sam"] = 'B';
	grade_list["satishcj"] = 'F'; // This will update the grade of Satish from A to
F
	map<string, char>::iterator map_it;
	map_it = grade_list.begin();
	cout << map_it->first;
	cout << map_it->second;
	map_it++;
	cout << map_it->first;
	cout << map_it->second;
	system("pause");
	return 0;
}
```

Printing all key and values in the map using iterator

```
#include<iostream>
	#include<string>
	#include<map>
	using namespace std;

	int main()
	{
		map<string, char> grade_list;
		grade_list["Satish"] = 'A';
		grade_list["Sam"] = 'B';
		grade_list["Rahul"] = 'F';
```

```
	map<string, char>::iterator map_it;
	for (map_it = grade_list.begin(); map_it != grade_list.end(); map_it++)
	{
		cout << (*map_it).first << (*map_it).second<<"\n";
	}
	system("pause");
	return 0;
}
```

Searching for a Value in the Map

```
#include<iostream>
	#include<string>
	#include<map>
	using namespace std;

	int main()
	{
		map<string, char> grade_list;
		grade_list["Satish"] = 'A';
		grade_list["Sam"] = 'B';
		grade_list["Rahul"] = 'F';
		map<string, char>::iterator map_it;
		for (map_it = grade_list.begin(); map_it != grade_list.end(); map_it++)
		{
			if((*map_it).first=="Satish")
			{
				cout << (*map_it).first << (*map_it).second << "\n";
			}
		}
		system("pause");
		return 0;
	}
```

Using the Find function for searching a value in the map

```
	#include<iostream>
	#include<string>
	#include<map>
	using namespace std;

	int main()
	{
		map<string, char> grade_list;
		grade_list["satish"] = 'A';
		grade_list["Sam"] = 'B';
		grade_list["satish"] = 'F';
		map<string, char>::iterator map_it;
		if (grade_list.find("satishcj") == grade_list.end())
		{
			cout << "Satish cj is not in the map";
		}
		else
		{
			cout << "Satish cj is available in the map";
		}
		system("pause");
```

```
        return 0;
}
```

Checking the size of the map

```
#include<iostream>
#include<string>
#include<map>
using namespace std;

int main()
{
        map<string, char> grade_list;
        grade_list["Satishcj"] = 'A';
        grade_list["Sam"] = 'B';
        grade_list["Rahul"] = 'F';
        cout << grade_list.size();
        system("pause");
        return 0;
}
```

Checking if the map is empty

```
#include<iostream>
#include<string>
#include<map>
using namespace std;

int main()
{
        map<string, char> grade_list;
        grade_list["Satishcj"] = 'A';
        grade_list["Sam"] = 'B';
        grade_list["Rahul"] = 'F';
        if (grade_list.empty())
        {
                cout << "The map is empty";
        }
        else
        {
                cout << "The map is not empty";
        }
        system("pause");
        return 0;
}
```

Removing a key from the map using erase function

```
#include<iostream>
#include<string>
#include<map>
using namespace std;

int main()
{
        map<string, char> grade_list;
        grade_list["Satishcj"] = 'A';
        grade_list["Sam"] = 'B';
        grade_list["Rahul"] = 'F';
        grade_list.erase("Satish");
        map<string, char>::iterator map_it;
        for (map_it = grade_list.begin(); map_it != grade_list.end(); map_it++)
        {
```

```
                cout << (*map_it).first << (*map_it).second;
        }
        system("pause");
        return 0;
}
```

Removing all elements from a map

```
 #include<iostream>
#include<string>
#include<map>
using namespace std;

int main()
{
        map<string, char> grade_list;
        grade_list["Satishcj"] = 'A';
        grade_list["Sam"] = 'B';
        grade_list["Rahul"] = 'F';
        grade_list.clear();
        if (grade_list.empty())
        {
                cout << "The grade list is empty";
        }
        system("pause");
        return 0;
}
```