Here is a sample implementation of class "Stack of Marks".

```
typedef int Mark;

class Marks
{
public:
	Marks() : marks(0), size(0) {}
	~Marks() { delete[] marks; }

	bool Empty() { return size; }
	int Size() { return size; }

	//Reference to an element at the top of the stack.
	Mark& Top() { return marks[size-1]; }

	//Adds an element to the top of the stack.
	void Push(const Mark & mark)
	{
		Mark * newMarks = new Mark[size+1];
		for(int i = 0; i < size; ++i) newMarks[i] = marks[i];
		newMarks[size] = mark;
		++size;
		delete[] marks;
		marks = newMarks;
	}

	//Removes the element from the top of the stack.
	void Pop()
	{
		Mark * newMarks = new Mark[size - 1];
		for (int i = 0; i < size-1; ++i) newMarks[i] = marks[i];
		--size;
		delete[] marks;
		marks = newMarks;
	}

	//Remove index-element.
	//Note, there is no such functionality in the standard stack
	void Delete(int index)
	{
		if (index <= 0 || index > size) return;
		Mark * newMarks = new Mark[size - 1];
		for (int i = 0; i < index - 1; ++i) newMarks[i] = marks[i];
		for (int i = index; i < size; ++i) newMarks[i] = marks[i];
		--size;
		delete[] marks;
		marks = newMarks;
	}

	// Display space separated marks
	//Note, there is no such functionality in the standard stack
	void Display()
	{
		for(int i = 0; i < size; ++i)
		{
			cout << marks[i] << " ";
		}
	}
private:
	Mark * marks; // pointer to dynamic array of marks
	int size;     // size of array
};
```