```
// stdafx.h : include file for standard system include files,
// or project specific include files that are used frequently, but
// are changed infrequently
//

#pragma once

#include "targetver.h"
#include <string>
#include <stdio.h>
#include <tchar.h>
#ifndef LIFO_H
#define LIFO_H
template <typename Type>
class LIFO{
private:
	enum {maxVelikost = 10};
	Type *prvky;
	int top;
	int velikostPole;
	void inity();
public:
	LIFO();
	LIFO(int max);
	bool isEmpty() { return top == 0; };
	bool isFull() { return top == velikostPole} ;
	bool push(const Type &item);
	bool pop(Type &prvek);
	void view();
	~LIFO();
};

template <typename Type>
LIFO<Type>::LIFO(int max) : velikostPole(max), top(0)
{
	prvky = new Type[velikostPole];
}



template <typename Type>
LIFO<Type>::LIFO()
{
	top = 0;
	velikostPole = maxVelikost;
	prvky = new Type[maxVelikost];
}

template <typename Type>
bool LIFO<Type>::push(const Type &item)
{
	if (top < velikostPole){
		prvky[top++] = item;
		return true;
	}else{
		return false;
	}
}

template <typename Type>
bool LIFO<Type>::pop(Type &prvek)
{

	if(top > 0){

		prvek =  prvky[--top];

		return true;
	}else{
		return false;
	}

}

template <typename Type>
void LIFO<Type>::view()
{
		for (int i = 0; i < top; i++){
			cout << prvky[i] << endl;
		}

}

template <typename Type>
LIFO<Type>::~LIFO()
{
	delete [] prvky;
	top = 0;
}

#endif
```

```
// SablonaLIFO.cpp : Defines the entry point for the console application.
//

#include "stdafx.h"
#include <iostream>
#include <string>
#include <cctype>
using namespace std;


int _tmain(int argc, _TCHAR* argv[])
{
	LIFO<int> zasobnik(5);
	// LIFO<int> zasobnik;

	int pom;

	zasobnik.push(10);
	zasobnik.push(20);
	zasobnik.push(30);
	zasobnik.push(40);
	zasobnik.push(50);
	zasobnik.push(60);
	zasobnik.push(70);



	if(zasobnik.isEmpty() == true){
		cout << "prazdny" << endl;
	}else {
		cout << "neni prazdny" << endl;
```

```
        }

        zasobnik.view();


        zasobnik.pop(pom);

        cout << "odebrany prvek: " << pom << endl;

        zasobnik.view();

        system("pause");
        return 0;
}
```