**Лабораторная работа № 13**

**Передача функций в качестве параметра.**

**Перезагрузка функций.**

**Создание проектов.**

**1. ЦЕЛЬ РАБОТЫ:**  приобретение практических навыков создания проектов.

**2. ОСНОВНЫЕ ПОНЯТИЯ**

Сначала создадим простейший проект, в который будет входить всего один файл.

• выбрать пункт верхнего меню *Файл – Создать – Проект*;

• в появившемся окне выбрать тип проекта *Empty Project* (пустой проект), ввести имя проекта и щелкнуть по кнопке *ОК*:

• теперь в левой части окна нужно включить вкладку *Проект*, где появится название проекта (папка); если нажать на правую кнопку мыши, появляется контекстное меню, в ко-

тором можно *Создать файл* (новый), *Добавить к проекту, Удалить из проекта*

Файл **main.cpp** практически повторяет старую программу, но не содержит процедур.

Тем не менее, в начале программы нужно вставить *объявления* всех используемых процедур (заголовки с точкой с запятой в конце). Этим мы скажем транслятору, что такие процедуры есть (в других файлах или в библиотеках), и определим типы их параметров.

Для глобальных объектов - В одном модуле (там, где выделяется память под глобальные переменные) они объявляются так же, как и обычно. 2. Во всех остальных модулях, использующих глобальные переменные, перед их объявлением ставится ключевое слово **extern** (от англ. *external –* внешний). Это означает, что они располагаются в каком-то другом модуле. **extern int X[];**Размеры массивов указывать не нужно, потому что память тут не выделяется. Начальные значения глобальных переменных можно задавать только в том модуле, где они размещаются в памяти (то есть объявляются без ключевого слова **extern**).

prog.c

*#include "func.h" или void sort(int \*x);*

*int m[]={8,2,4,1,6,5};*

*int N=sizeof(m)/sizeof(int);*

*int main() {*

*sort(m);*

*for(int i=0; i<N; i++) cout<<m[i]<<" ";*

*cout<<endl<<N<<endl;*

*getch();*

*}*

func.c

*extern int N;*

*void swap(int &x, int &y) {*

*int t=x; x=y; y=t;*

*}*

*void sort(int \*x) {*

*int i, k;*

*do {*

*k=0;*

*for(i=1; i<N; i++)*

*if(x[i]<x[i-1]) { swap(x[i], x[i-1]); k=1;}*

*} while(k);*

*cout<<N<<endl;*

*}*

func.h

*void swap(int &x, int &y);*

*void sort(int \*x);*

**Передача функции в качестве параметра.**

#include <iostream.h>

int xx (int a) {

cout <<a<<"+"<<a<<endl;

return a+a;

}

int yy ( int a) {

cout <<a<<"\*"<<a<<endl;

return a\*a;

}

int call (int a, int (\*f) (int)) {

cout <<"call " << " f() = " << f(a) <<"\n";

}

int main () {

call(3,xx);

call(4,yy);

}

**Перезагрузка функций**

Используется одно имя, но параметры различны по количеству или типу.

### int add(int a,int b) {

### return(a + b);

### }

### int add (int a, int b, int c) {

### return(a + b + c);

### }

### int add (int a, float b) {

### return(b+a);

### }

### main(void)

### {

### cout << add(200, 101) << endl;

### cout << add(100, 201, 700) << endl;

### cout << add(700.1,1) << endl;

### }

**3. ВЫПОЛНЕНИЕ РАБОТЫ**

Создать проект, для оценки времени работы различных функций сортировки массивов различных типов (целых и вещественных). Функции сортировки разместить в библиотеке.

Вызов фунции оценки времени сортировки должен выглядеть как:

time=timer(tip\_sort(a, n));

где а – массив для сортировки, который может быть целого или вещественного типа.

n-кол элементов в массиве.

**n взять достаточно большим для оценки времени сортировки**