

Структура программы

#директивы препроцессора

.....

#директивы препроцессора

функция а ()

операторы

Определение прототипов функций

Определение глобальных переменных

void main () //функция, с которой начинается выполнение программы

операторы

описания

присваивания

вызов функции

пустой оператор

составной

выбора

циклов

перехода

функция в ()

операторы

Препроцессорные директивы

- ▶ Начинаются в символа #
- ▶ 1. Подключение заголовочных файлов

Формат:

```
#include <ID_файла>
```

<> - поиск в стандартной директории

"" – поиск в текущей директории

stdio

iostream

cmath

▶ 2. Обработка макроопределений

▶ `#define <ID> <строка>`

▶ `#define SIZE 100`

Функция main

- ▶ Управление всей работой проекта

```
int _tmain(int argc, _TCHAR* argv[])  
{  
  
}
```

stdafx.h

stdafx.cpp

Декларация объектов

- ▶ Объявление - описание не приводящее к выделению памяти

<класс памяти> <тип> <список объектов>

- ▶ Определение , при кот для объекта выделяется объем памяти и его можно инициализировать – задать начальное значение

тип имя_переменной = начальное_значение;

Время существования и область видимости переменных

Областью действия

- блок операторов (`{...}`);
- модуль (файл);
- вся программа в целом.

Временем жизни

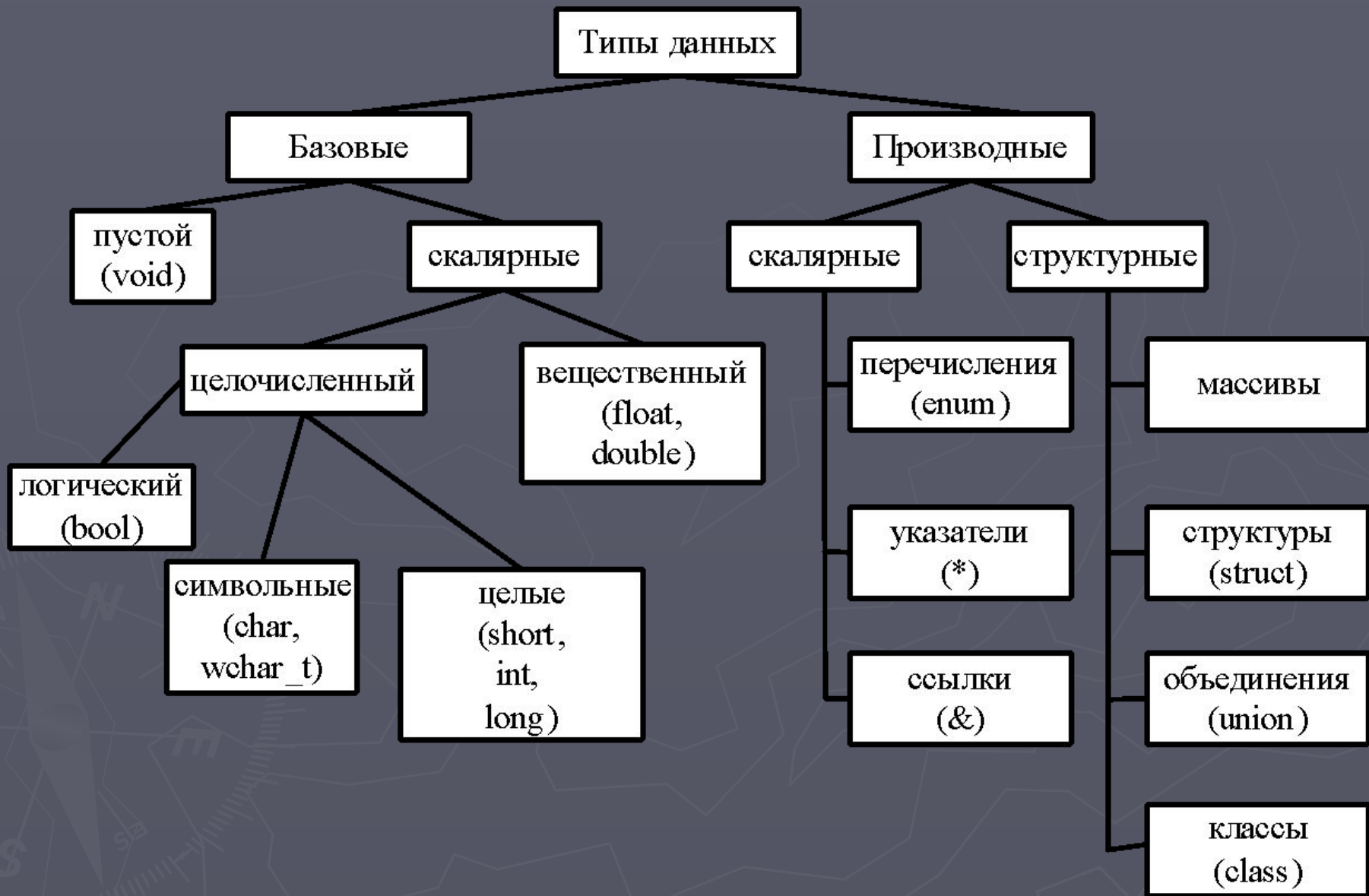
- Локальное
- глобальное

<класс памяти>

- ▶ Способ размещения объекта в памяти определяет область видимости и время жизни переменной
- ▶ По умолчанию auto

Спецификаторы класса памяти

- ▶ auto
- ▶ static
- ▶ extern
- ▶ register



<limits.h>

ОПЕРАТОРЫ ВВОДА-ВЫВОДА

- ▶ ПОТОКОВЫЙ ВВОД И ВЫВОД
- ▶ ФОРМАТИРОВАННЫЙ ВВОД И ВЫВОД
- ▶ СТРОКОВЫ И СИМВОЛЬНЫЙ ВВОД И ВЫВОД

ПОТОКОВЫЙ ВВОД И ВЫВОД

- ▶ `#include <iostream>`
- ▶ `cout <<` по умолчанию к монитору
- ▶ `cin >>` к клавиатуре
- ▶ `cerr` с ограниченной буферизац.

манипуляторы

- ▶ `setfill` - установит заполнение
`setfill('S');`
- ▶ `setw` - ширина поля вывода
- ▶ `endl` – конец линии
- ▶ `setbase` - установить основание
`cout << setbase(10);`
- ▶ `setprecision` - Точность для дробных
`setprecision(3);`
`#include <iomanip>`

ANSI/ISO C++

- 1) `std::cout` (детально)
- 2) `using std::cout;` (использовать объявление)
- 3) `using namespace std;` (использовать напрямую)

```
std::cout << "Hello ";
```

```
using namespace std;  
cout << "World." << endl;
```

Форматированный ввод и вывод

```
#include <stdio.h>
```

функция вывода информации

```
printf ( <форматная строка> , <список аргументов> );
```

```
printf ( "Значение числа Пи равно %f\n", pi);
```

```
printf ( "Значение числа Пи равно %f\n", pi);
```

%d - десятичное целое число;

%f - вещественное число типа float или double;

%c - символ;

%s - строка;

%p - указатель;

%u – беззнаковое целое число;

%o – целые числа в восьмеричной системе счисления;

%x – целые числа в шестнадцатеричной системе счисления;

%e – вещественное число в экспоненциальной форме.

после % цифра – минимальная ширина поля ввода

	%10d	%7.2f
--	------	-------

Управляющие символы

- ▶ `\n` - перемещает курсор в начальную позицию следующей строки;
- ▶ `\t` – перемещает курсор в следующую позицию табуляции экрана;
- ▶ `\r` – выполняет «возврат каретки», перемещая курсор к началу той же строки без перехода на следующую;
- ▶ `\b` – передвигает курсор только на одну позицию влево.

функция ввода информации

```
scanf ( <форматная строка>, <список  
аргументов> );
```

форматная строка – модификаторы
форматов, тип и порядок которых должен
совпадать с объектами

список аргументов – адреса переменных
разделенные запятыми

& - взять адрес

```
int cours;  
char name[20];  
printf ("Укажите курс и имя \n");  
scanf ("%d%s", &cours, name);
```

Строковые и символный ввод и вывод

```
#include <stdio.h>
```

```
puts("привет!");
```

```
putchar('H');
```

```
char name [60];
```

```
printf("Как вас зовут: ");
```

```
gets (name);
```

```
printf ("Привет, %s\n", name);
```

```
int letter;  
letter = getchar();
```



Пишем по-русски

```
#include <locale>
{...
setlocale(LC_STYPE, "Russian");
// ИЛИ
setlocale(LC_ALL, "rus");
```

```
_getch();
```

