Модульность — фундаментальный аспект всех успешно работающих крупных систем.

Б. Страуструп

Директивы препроцессора

- ✔ Директива #include.
- ✔ Директива #define.
- ✔ Директива #undef.

МОДУЛЬНОСТЬ В ЯЗЫКЕ С++ ПОДДЕРЖИВАЕТСЯ С ПОМОЩЬЮ:

- *Директив препроцессора.*
- *Пространств имен.*
- *Классов памяти.*
- *Исключений*

ПРЕПРОЦЕССОР.

- *Препроцессором называется первая фаза компилятора.*
- *Инструкции препроцессора называются* ***директивами.***

Директивы должны начинаться с символа **#**, *перед которым в строке могут находиться только пробельные символы.*

Директива
#include.

ДИРЕКТИВА #INCLUDE.

Директива **#include <имя_файла>** *вставляет содержимое указанного файла в ту точку исходного файла, где она записана.*

- *Включаемый файл также может содержать директивы* **#include**.

Поиск файла, если не указан полный путь, ведется в стандартных каталогах включаемых файлов.

- *Вместо угловых скобок могут использоваться кавычки (" ") — в этом случае поиск файла ведется в каталоге, содержащем исходный файл, а затем уже в стандартных каталогах.*

Директива **#include** *является простейшим средством обеспечения согласованности объявлений в различных*

ЗАГОЛОВОЧНЫЕ ФАЙЛЫ.

Заголовочные файлы обычно имеют расширение **.h** *и могут содержать:*

- *определения типов, констант, встроенных функций, шаблонов, перечислений;*
- *объявления функций, данных, имен, шаблонов;*
- *пространства имен;*
- *директивы препроцессора;*
- *комментарии.*

В заголовочном файле не должно быть определений функций и данных.

- *Эти правила не являются требованием языка, а отражают разумный способ использования директивы.*

При указании заголовочных файлов стандартной библиотеки расширение **.h** можно опускать.

- *Старые версии компиляторов могут не поддерживать это требование стандарта.*

Заголовочные файлы.

Для каждого файла библиотеки C с именем **<name.h>** *имеется соответствующий файл библиотеки* **C++ <cname>**, *в котором те же средства описываются в пространстве имен* **std**.

- *Например, директива* **#include <cstdio>** *обеспечивает те же возможности, что и* **#include <stdio.h>**, *но при обращении к стандартным функциям требуется указывать имя пространства имен* **std**.

Заголовочные файлы.

Директива
#define.

ДИРЕКТИВА #DEFINE.

Директива **#define** *определяет подстановку в тексте программы.*

Она используется для определения:

- ***символических констант:***

  #define имя текст_подстановки

  (все вхождения имени заменяются на текст подстановки);

- ***макросов****, которые выглядят как функции, но реализуются подстановкой их текста в текст программы:*

  #define имя(параметры) текст_подстановки

- ***символов, управляющих условной компиляцией****.*

Формат: **#define имя**

Примеры:

```
#define VERSION 1
#define VASIA "Василий Иванович"
#define MAX(x,y)  ((x)>(y)?(x):(y))
#define MUX
```

Имена рекомендуется записывать прописными буквами

- *чтобы зрительно отличать их от имен переменных и функций*

Директива #define.

```
#define MAX(x,y)  ((x)>(y)?(x):(y))
```

Параметры макроса используются при макроподстановке, например, если в тексте программы используется вызов макроса:

```
y = MAX(sum1, sum2);
```

он будет заменен на:

```
y = ((suml)>(sum2)?(suml):(sum2));
```

Директива #define.

Отсутствие круглых скобок может привести к неправильному порядку вычисления

- *поскольку препроцессор не оценивает вставляемый текст с точки зрения синтаксиса.*

Например:

если к макросу **#define sqr(x) (x*x)**
обратиться как **sqr(y+1)**

в результате подстановки получится выражение:

(y+1*y+1)

Директива #define.

Макросы и символические константы унаследованы из языка C, при написании программ на C++ их следует избегать.

Вместо символических констант предпочтительнее использовать **const** *или* **enum**, *а вместо макросов — встроенные функции или шаблоны.*

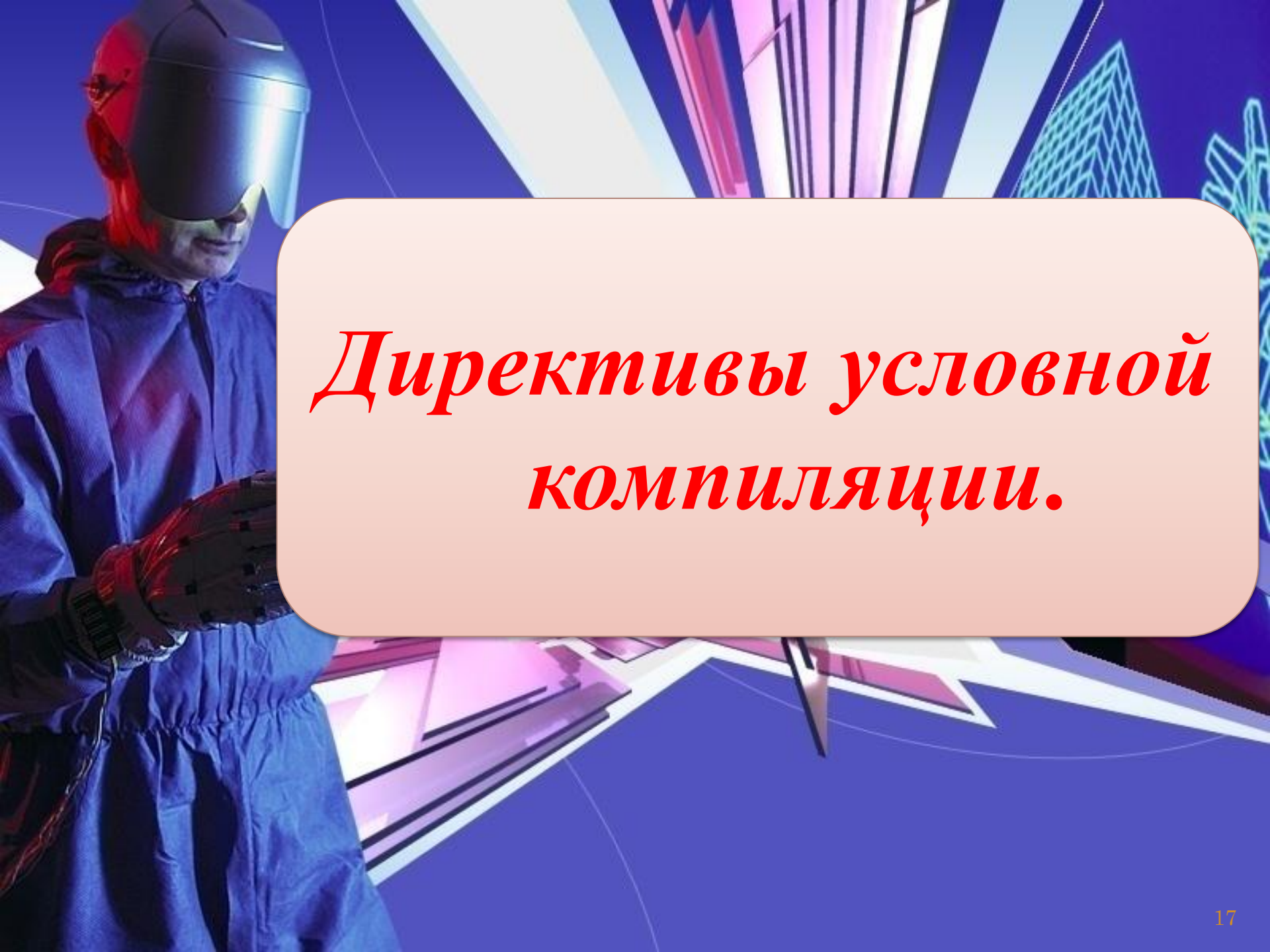
Директива #define.

Директива

#undef.

ДИРЕКТИВА #UNDEF.

- *Директива* **#undef** *удаляет определение символа.*
- *Используется редко, например, для отключения какой-либо опции компилятора.*
- *Формат:* **#undef имя**



Директивы условной компиляции.

ДИРЕКТИВЫ УСЛОВНОЙ КОМПИЛЯЦИИ.

Директивы условной компиляции **#if**, **#ifdef** *и* **#ifndef** *применяются для того, чтобы исключить компиляцию отдельных частей программы.*

Формат :

```
#if константное_выражение
        ......
    [ #elif константное_выражение
        ......]
        [ #elif константное_выражение
            ......]
    [ #else
        ......]
#endif
```

Количество директив **#elif** *— произвольное.*

Директива #if.

Пример:

```
// Пример условного включения
// различных версий заголовочного файла
#if VERSION == 1
        #define INCFILE   "vers1.h"
    #elif VERSION == 2
            #define INCFILE   "vers2.h"
                /* и так далее */
    #else
        #define INCFILE   "versN.h"
#endif
#include INCFILE
```

В константных выражениях может использоваться проверка, определена ли константа, с помощью **defined (имя_константы)**.

Директива #if.

Формат :

```
#ifdef символ
// Расположенный ниже код компилируется,
// если символ определен

#ifndef символ
// Расположенный ниже код компилируется,
// если символ не определен
```

Действие этих директив распространяется до первого **#elif, #else** *или* **#endif**.

Директивы #ifdef и #ifndef.

Пример:

```
// Директива #ifndef часто применяется для того,
// чтобы обеспечить включение заголовочного
// файла только один раз
#ifndef HEADER_INCLUDED
        #include "myheader.h"
        #define HEADER_NCLUDED
#endif
```

Директива #ifdef.

Предопределенные макросы

- ***В C++ определено несколько макросов, предназначенных в основном для того, чтобы выдавать информацию о версии программы или месте возникновения ошибки.***

_cplusplus

- *определен, если программа компилируется как файл* C++.
- *применяется, если требуется переносить код из* C *в* C++ *и обратно.*

Многие компиляторы при обработке файла с расширением **.c** считают, что программа написана на языке C.

Использование этого макроса позволяет указать, что можно использовать возможности C++:

```
#ifdef _cplusplus
// Действия, специфические для C++
#endif
```

Предопределенные макросы.

DATE	• текущей датой в формате
FILE	• полным именем
LINE	• текущая строка исходного текста.
TIME	• текущее время.

```
printf("Ошибка в файле %s \n", _FILE_);
printf(" Дата компиляции - %s \n", _DATE_);
printf(“ Время компиляции:%s\n ", _FILE_, _TIME_);
```

Предопределенные макросы.

Пример:

```
// Директива #ifndef часто применяется для того,
// чтобы обеспечить включение заголовочного
// файла только один раз
#ifndef HEADER_INCLUDED
        #include "myheader.h"
        #define HEADER_NCLUDED
#endif
```

Предопределенные макросы.