# Лабораторная работа № 1

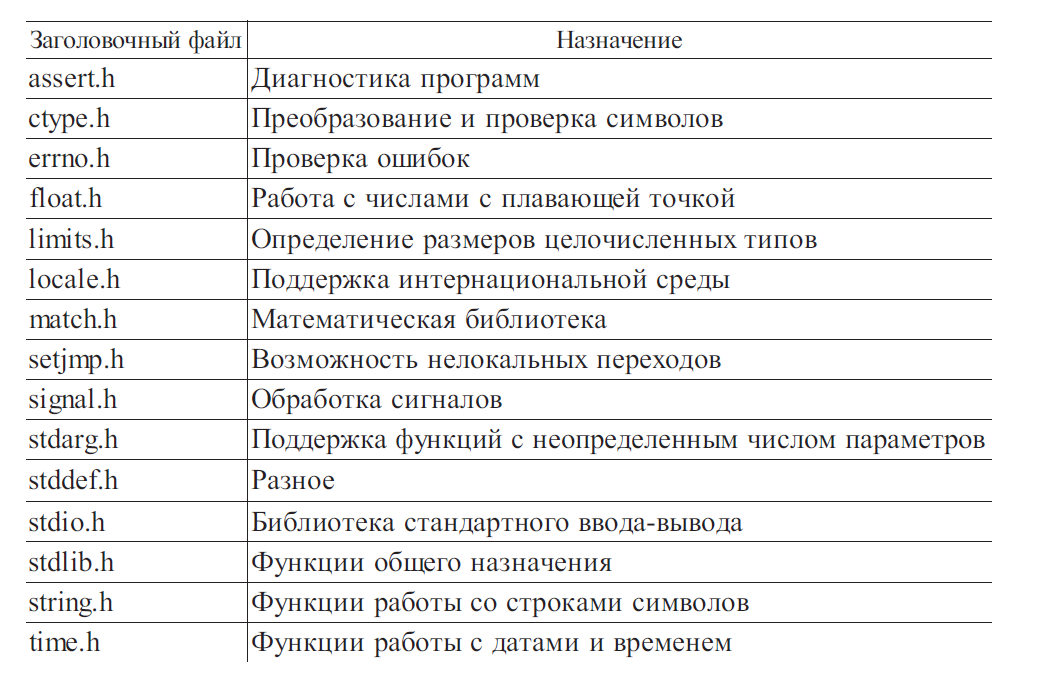
# Синтаксис языка СИ.

## ***Что такое препроцессор, его функции и обращение к нему. Опишите основные библиотеки и их назначение.***

Препроцессор Си - это программа, которая обрабатывает входные данные для компилятора. Для препроцессора предназначены строки программы, начинающиеся с символа #. В одной строке разрешается записывать только одну команду (директиву препроцессора).

Основные функции:

* Подключение к исходной программе заданных файлов
* Осуществление подстановок
* Управление условиями компиляции

Основные библиотеки :

Сколько функций минимально должна содержать программа на С?

Как минимум одну - функцию main.

## Что реализует функция main?

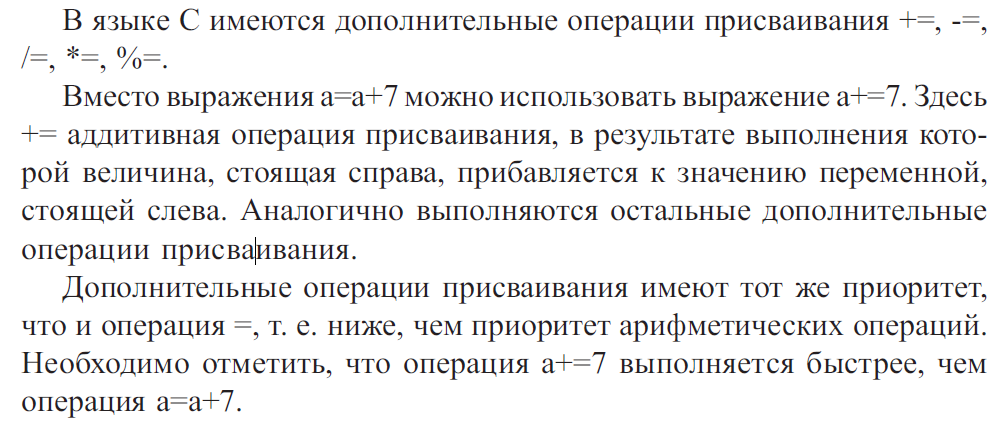
Функция main вызывает запуск программы и содержит другие функции. Это начальная точка выполнения для всех программ на языках C

## Напишите способы задания комментария в программе и укажите на их отличия.

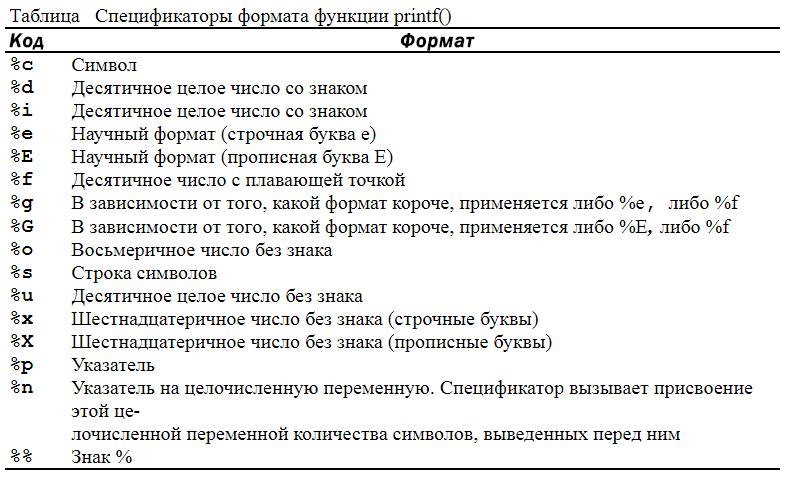
Текст, заключённый в служебные символы /\* и \*/ является многострочным комментарием. Компиляторы, совместимые со стандартом C99, также позволяют использовать строчные комментарии, начинающиеся с символов // и заканчивающиеся переводом строки.

## Оператор присваивания.

Оператором присваивания является “=”.



## Оператор вывода. Формат вывода.

Вывода на печать осуществляется с помощью функции printf(). Формат вывода задаётся с помощью символа «%». До операции ввода-вывода необходимо включение внешнего файла – stdio.h, в котором содержатся данные функции. 

Специальные символы, которые используются в выводе и их назначение.

* '\n' — перевод строки;
* '\t' — горизонтальная табуляция;
* '\v' — вертикальная табуляция;
* '\b' — возврат на символ;
* '\r' — возврат на начало строки;
* '\a' — звуковой сигнал.

## Результаты выполненной работы:

int dir( )/\* dir\*/ // описание функции dir и комментарий с ее именем

{ // открывающая операторная скобка

printf("добрый день\n"); //вывод текста “добрый день” и переход на новую строчку

printf(" мы рады вас видеть\n"); //вывод текста “рады вас видеть” и переход на новую строчку

} //закрывающая операторная скобка

int main () // описание функции main, начало программы

{ //открывающая операторная скобка

dir (); //вызов функции dir

printf("надеемся на плодотворную работу \n"); // вывод текста “надеемся на плодотворную работу ” и переход на новую строчку

printf("и замечательные успехи\n"); // вывод текста замечательные успехи ” и переход на новую строчку

} //закрывающая операторная скобка

## Описать все компоненты строки программы printf("сейчас %d год \n", year);.

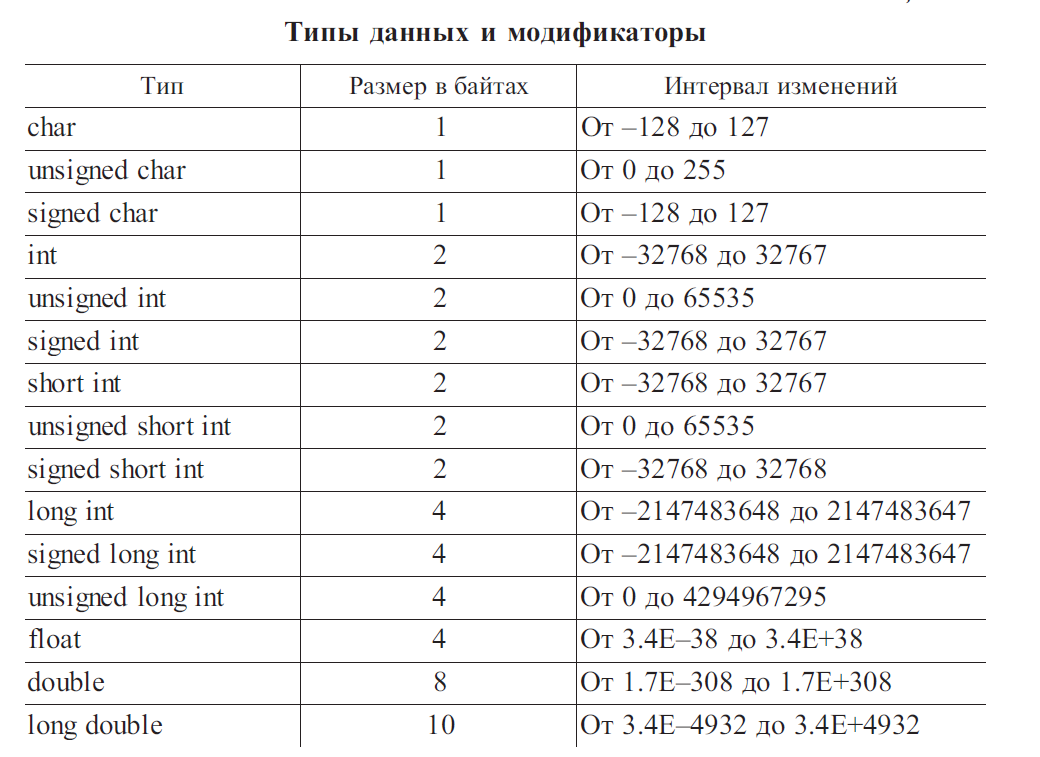
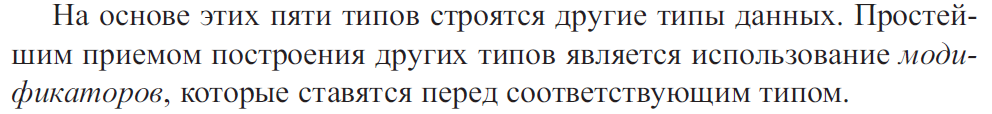
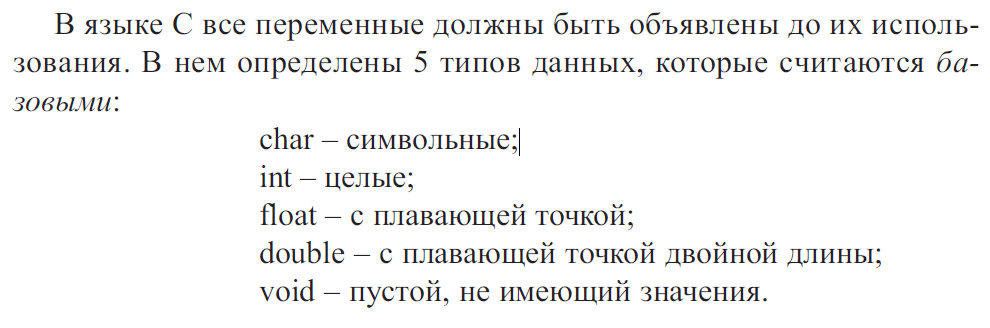
printf – оператор вывода;

%d – вид выводимой на экран информации;

\n – перевод строки;

year – аргумент функции printf;

Какие типы данных существуют. Их обозначение. Запись переменных.



Структуры в Си позволяют хранить несколько полей и в одной переменной. В других языках могут называться записями или кортежами. Например, данная структура хранит в себе имя человека и дату рождения:

struct birthday

{

char name[20];

int day;

int month;

int year;

};

Объявление структур в теле программы всегда должно начинаться с ключевого struct (необязательно в C++). Доступ к элементам структуры осуществляется с помощью оператора . или ->, если мы работаем с указателем на структуру. Структуры могут содержать указатели на самих себя, что позволяет реализовывать многие структуры данных, основанных на связных списках. Такая возможность может показаться противоречивой, однако все указатели занимают одинаковое число байт, поэтому размер этого поля не изменится от числа полей структуры.

Структуры не всегда занимают число байт, равное сумме байт их элементов. Компилятор обычно выравнивает элементы в блоки по 4 байта. Также есть возможность ограничить число бит, отводимое на конкретное поле, для этого надо после имени поля через двоеточие указать размер поля в битах. Такая возможность позволяет создавать битовые поля.

Некоторые особенности структур:

* Адрес памяти первого поля структуры равен адресу самой структуры
* Структуры могут быть инициализированы или приведены к какому-либо значению, с помощью составных литералов
* Пользовательские функции могут возвращать структуру, хотя часто не очень эффективны во время выполнения. С C99, структура может оканчиваться массивом переменного размера.
* Имя переменной (идентификатор переменной) - это буквенно-цифровая последовательность символов алфавита, начинающаяся с буквы.

Указатель — адресная переменная, содержащая адрес объекта. Указатель не несет информации о содержимом объекта, а содержит сведения о том, где размещен объект. Указатели широко используются в программировании на языке Си.Указатели часто используются при работе с массивами.Память компьютера можно представить в виде последовательности пронумерованных однобайтовых ячеек, с которыми можно работать по отдельности или блоками.Каждая переменная в памяти имеет свой адрес - номер первой ячейки, где она расположена, а также свое значение. Указатель — это тоже переменная, которая размещается в памяти. Она тоже имеет адрес, а ее значение является адресом некоторой другой переменной. Переменная, объявленная как указатель, занимает 4 байта в оперативной памяти (в случае 32-битной версии компилятора).Указатель, как и любая переменная, должен быть объявлен.

Общая форма объявления указателя

тип \*ИмяОбъекта;

Тип указателя— это тип переменной, адрес которой он содержит.

Массив - это непрерывный участок памяти, содержащий последовательность объектов одинакового типа, обозначаемый одним именем.

Массив характеризуется следующими основными понятиями:

Элемент массива (значение элемента массива) – значение, хранящееся в определенной ячейке памяти, расположенной в пределах массива, а также адрес этой ячейки памяти.

Каждый элемент массива характеризуется тремя величинами:

адресом элемента - адресом начальной ячейки памяти, в которой расположен этот элемент;

индексом элемента (порядковым номером элемента в массиве);

значением элемента.

Адрес массива – адрес начального элемента массива.

Имя массива – идентификатор, используемый для обращения к элементам массива.

Размер массива – количество элементов массива

Размер элемента – количество байт, занимаемых одним элементом массива.

Графически расположение массива в памяти компьютера можно представить в виде непрерывной ленты адресов.

Длина массива – количество байт, отводимое в памяти для хранения всех элементов массива.

ДлинаМассива = РазмерЭлемента \* КоличествоЭлементов

Для определения размера элемента массива может использоваться функция

int sizeof(тип);

Например,

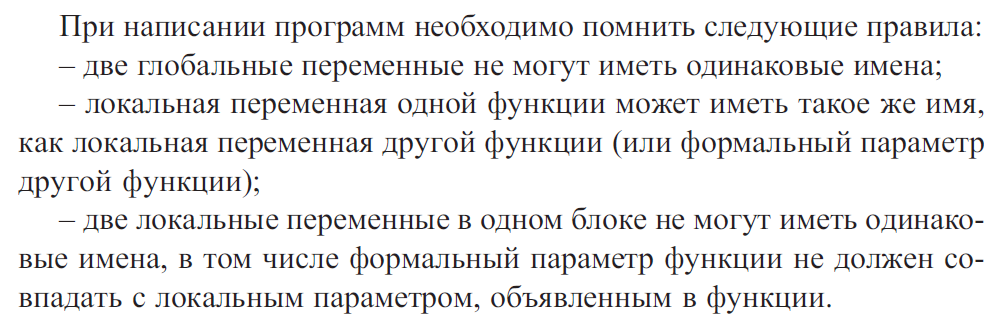
sizeof(char) = 1;

sizeof(int) = 4;

sizeof(float) = 4;

sizeof(double) = 8;

Кроме букв и цифр для образования имени переменной можно также использовать символ подчеркивания(\_), который в этом случае рассматривается как буква, т.е. может быть первым в последовательности символов, образующих имя.

На длину имени не накладывается ограничений, однако для компилятора Си значащими являются первые 32 символа и, таким образом в имя переменной легко вносить содержательный смысл.

## Назовите ошибку в имени идентификатора 40Hours, Get Data, box-22, cost\_in\_$, int.

40Hours – первый символ не является буквой;

Get Data – пробел;

box-22 – недопустимый символ “-“;

cost\_in\_$ - недопустимый символ ”$”;

int – название переменной совпадает с ключевым словом;

## Есть ли разница в идентификаторах Hours и hours (машина воспримет как две переменных или как одну и ту же)?

Машина воспримет как две переменные.

## Список идентификаторов (обозначение переменных):

|  |  |
| --- | --- |
| Переменная | Выражение |
| Alpha | 2856 |
| Num | 2856, alpha |
| Rate | 0.36 |
| Ch | ‘b’ |

## Код программы:

* Сложение “+”;
* Вычитание “–“;
* Умножение “\*”;
* Деление и целочисленное дение “/”;
* Остаток от деления “%”;
* Возведение в степень

## #include < math.h >

## double pow (double x, double y);

## float powf (float x, float y);

## long double powl (long double x, long double y);

## Операции инкремента и декремента в C.

Увеличение на единицу (инкрементация) “++”

Уменьшение на единицу (декрементация) “--“

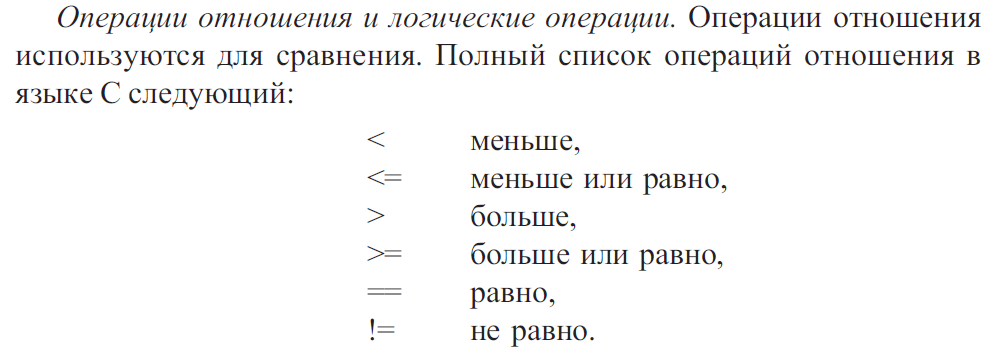
## Получить результат выражения b = (a++)\*5; (при a=4).

Ответ: b= 20;

## Получить результат выражения b = (++a)\*5; (при a=4).

Ответ: b= 25;

Операции отношения (сравнения): равно, не равно, больше, меньше, больше или равно, меньше или равно.



## Логические операции: не, и, или.

## D:\YandexDisk\Скриншоты\2018-02-08_22-58-15.png

## Оператор ввода с клавиатуры:

Для ввода с клавиатуры используется функция scanf.

## Библиотечные функции:

abort - сбросить выполнение программы

abs - абсолютное значение целого

acos - арккосинус

asctime - дать время

asin - арксинус

assert - проверить утверждение

atan, atan2 - арктангенс

atof, atoi, atol - преобразовать в плавающее

bdos, bdosx - вызвать функцию dos

bioskey, \_bios\_keybrd - работа с клавиатурой

bsearch - двоичный поиск в массиве

calloc - дать память

ceil - целая верхняя грань

chdir - сменить текущий каталог

\_chkstack - проверить стек

clearerr - сбросить флаг ошибки

clock - определить процессорное время

close - закрыть файл

cos, cosh - косинус, косинус гиперболический

creat - создать файл

ctime - дать календарное время

difftime - определить отрезок времени

Display - package пакет отображения

div - делить

DOS - package пакет функций dos

ecvt - преобразовать число в строку

execl - execvp - запустить процесс

exit, \_exit - завершить выполнение программы

exp - экспонента

fabs - абсолютное значение вещественного

farcalloc - farrealloc - работа с дальней кучей

\_farptr\_norm - нормализовать дальний указатель

\_farptr\_fromlong - дальний указатель из длинного

\_farptr\_tolong - дальний указатель в длинное

fclose - закрыть файл

fcvt - преобразовать double в строку с фиксированной точкой

feof - проверка признака конца файла

ferror - проверка признака ошибки в файле

fflush - вытолкнуть буфер файла

fgetc - взять байт из файла

fgets - взять строку из файла

fileno - дать дескриптор файла

filesize - дать размер файла

findfirst, findnext - поиск файлов по шаблону

floor - целая часть

flushall - вытолкнуть буфера всех открытых файлов

fmod - остаток от деления двух чисел

fopen - открыть файл

FP\_OFF, FP\_SEG - сегмент, смещение

fprintf - форматный вывод в файл

fputc - запись байта в файл

fputs - запись строки в файл

fread - читать из файла

free - освободить память

freopen - открыть файл повторно

frexp - экспоненциальное представление

fscanf - форматный ввод из файла

fseek - позиционировать файл

fstat - взять статус файла

ftell - дать позицию в файле

fwrite - писать в файл

getc, getchar, getche, getch - взять байт из файла

getcwd - дать текущий каталог

getDS - взять значение регистра ds

getenv - дать окружение

gets - ввод строки

hypot - гипотенуза

index - найти литеру в строке

inp, inpw - ввод из порта

Interrupt - package пакет работы с прерываниями

int86, int86x - программное прерывание

intdos, intdosx - системный вызов функции dos

isatty - определить тип потока

is - package пакет классификации литер

itoa - представление целого

kbhit - проверка ввода с клавиатуры

labs - абсолютное значение длинного

ldexp - экспонента

ldiv - деление чисел типа long

localtime - дать местное время

log, log10 - логарифм

longjmp - длинный переход

lseek - изменить позицию в файле

malloc - отвести память

matherr - обработка ошибки математической функции

memchr - memset - работа с байтами в массивах

mkdir - создать каталог

MK\_FP - создать дальний указатель

mktime - преобразовать время

modf - дробная и целая часть числа

Mouse - package пакет работы с мышью

open - открыть файл

outp, outpw - вывод в порт

peek - переслать буфер издалека

perror - выдать сообщение об ошибке

poke - переслать буфер далеко

poly - полином

pow - степень

printf, fprintf, sprintf - форматный вывод

putc, putchar - вывод литеры в файл

puts - вывод строки в файл

qsort - быстрая сортировка таблицы

raise - возбудить сигнал

rand - случайная величина

read - читать блок из файла

realloc - переразместить блок памяти

rename - переименовать файл

rewind - установить указатель файла в начало

rmdir - удалить каталог

sbrk - увеличить сегмент данных

scanf - форматный ввод

segread - читать сегментные регистры

setbuf - установить буфер в/в

setjmp - установить точку длинного перехода

setvbuf - установить буфер в/в

signal - установить реакцию на сигнал

sin, sinh - синус, синус гиперболический

Sound - package пакет работы со звуком

spawn - создать процесс

sprintf - форматный вывод в буфер

sqrt - квадратный корень

srand - инициализация случайной величины

sscanf - форматный вывод из буфера

stat - читать статус файла

strcat - strdup - работа со строками

strerror - получить информацию об ошибке

strlen - дать длину строки

strlwr - привести к нижнему регистру

strncat - strpbrk - работа со строками

strrchr - strset - работа со строками

strspn - дать длину совпадающей подстроки

strstr - найти подстроку

strtod - strtoul - преобразовать из а/ц вида

strupr - привести к верхнему регистру

swab - попарно поменять байты

system - обращение к dos

tan, tanh - тангенс, тангенс гиперболический

time - дать время

toascii, tolower, toupper - b ascii,

ungetc - возвратить литеру

unlink - удалить файл

utime - изменить время модификации файла

va\_arg - va\_start - работа с переменным числом параметров

vprintf, vfprintf, vsprintf - форматный вывод

write - писать в файл;

## Вычислить отношение длинны окружности к площади круга:

#include <stdio.h>

#include <math.h>

int main()

{

float dlina, ploshad, otnoshenie;

printf("Vvedite dliny okruzhnosti ");

scanf("%f", &dlina);

ploshad= (dlina\*dlina)/(4\*(M\_PI));

otnoshenie=dlina/ploshad;

printf("Ploshad = %f, Otnoshenie = %f", ploshad, otnoshenie);

}

Напишите программу, позволяющую понять различие операции ++а и а++:

#include <stdio.h>

int main()

{

int a=10;

int b=70;

a++;

++b;

printf("a=%d b=%d\n",a,b);

printf("a=%d b=%d\n",a++,++b);

}