Семинар 1. Лексические основы, арифметические типы данных, переменные и константы, операции, линейный алгоритм

3. Операции, линейный алгоритм

1. Классификация операций
2. Описание операций
3. Математические функции
4. Линейный алгоритм

3.1. Классификация операций

По количеству операндов, операции:

- унарные (операнд один), которые, в свою очередь, по порядку записи операнда и операции делятся на:
  - префиксные
  - постфиксные
- бинарные (два операнда)

3.1. Классификация операций

По назначению, операции:

- аддитивные
- мультипликативные
- сдвига
- сравнения
- логические поразрядные
- логические бинарные
- присваивания
- тернарная операция

3.1. Классификация операций

По назначению, операции:

- приведения типов
- доступа к компонентам объектов
- генерации исключения (throw)
- вычисления размера объекта (sizeof)
- идентификации типа (typeid)
- выделения/освобождения памяти (new/delete)
- запятая

3.2. Аддитивные операции

- унарный инкремент
- унарный декремент
- унарный плюс
- унарный минус
- бинарный плюс
- бинарный минус

3.2. Аддитивные операции


```
#include <iostream>
using namespace std;
int main()
{
  int i = 0;
  cout << "i = " << i << endl;
  cout << "i++ = " << i++ << endl;
  cout << "++i = " << ++i << endl;
  cout << "i+i = " << i+i << endl;
  cout << "i-- = " << i-- << endl;
  cout << "--i = " << --i << endl;
  cout << "i = " << i << endl;
  return 0;
}
```

3.2. Аддитивные операции

Вывод:

i = 0

i++ = 0

++i = 2

i+i = 4

i-- = 2

--i = 0

i = 0

3.2. Мультипликативные операции

*	Умножение
/	Деление
%	Получение остатка от деления

-20/3 равняется -6

13%4 равняется 1

3.2. Операции сдвига

<<	Сдвиг влево двоичного представления значения левого операнда на кол-во разрядов, равного значению правого целочисленного операнда
>>	Сдвиг вправо

3.2. Операции сдвига

4 << 1 равняется 8

($4_{10}=100_2$, $8_{10}=1000_2$)

10 >> 2 равняется 2

($10_{10}=1010_2$, $2_{10}=10_2$)

3.2. Операции сравнения

==	Равно
!=	Не равно
>	Больше
<	Меньше
>=	Больше или равно
<=	Меньше или равно

3.2. Аддитивные операции


```
#include <iostream>
using namespace std;
int main()
{
  int i = 8;
  cout << "1 == 2 is " << (1 == 2) << endl;
  cout << "-100 < 100 is " << (-100 < 100) << endl;
  return 0;
}
```

Вывод:

```
1 == 2 is 0
-100 < 100 is 1
```

3.2. Логические операции

&	Поразрядная конъюнкция
|	Поразрядная дизъюнкция
^	Поразрядное исключающее или
~	Поразрядное отрицание
!	Логическое отрицание
&&	Конъюнкция
||	Дизъюнкция

3.2. Операция присваивания

Оператор присваивания в Си++ записывается как «=».

Операция присваивания выполняется справа налево.

3.2. Операция присваивания

Пример. Обмен значений двух переменных без участия третьей:

```
int a = 10;
int b = 3;
a = a + b;
b = a - b;
a = a - b;
cout << "a = " << a << endl; // a = 3
cout << "b = " << b << endl; // b = 10
```

3.2. Операция присваивания

*=	a *= b эквивалентно a = a * b
/=	a /= b эквивалентно a = a / b
%=	a %= b эквивалентно a = a % b
+=	a += b эквивалентно a = a + b
-=	a -= b эквивалентно a = a - b
<<=	a <<= b эквивалентно a = a << b
>>=	a >>= b эквивалентно a = a >> b

3.2. Операция присваивания

&=	a &= b эквивалентно a = a & b
|=	a |= b эквивалентно a = a | b
^=	a ^= b эквивалентно a = a ^ b

3.2. Операции приведения типов

(тип)операнд	Унаследована из языка Си. Не изменяя самого операнда, операция преобразует его значение к типу
тип(операнд)	Функциональная форма преобразования типа. Может использоваться только в тех случаях, когда тип имеет несоставное наименование.

int(true) эквивалентно 1

bool(2) эквивалентно true

char(51) эквивалентно 3

3.2. Операции приведения типов

dynamic_cast<тип> (выражение)	Приведение типов с проверкой допустимости приведения во время выполнения программы
static_cast<тип> (выражение)	Эквивалентно тип(выражение) для базовых классов
reinterpret_cast<тип> (выражение)	Приведение типов без проверки допустимости
const_cast<тип> (выражение)	Аннулирует действие модификатора const

3.3. Математические функции

Для использования математических функций необходимо подключить библиотеку math.h: include <math.h>

https://ru.wikipedia.org/wiki/Math.h

3.3. Математические функции

abs	Возвращает абсолютную величину целого числа
acos	арккосинус
asin	арксинус
atan	арктангенс
atan2	арктангенс с двумя параметрами
ceil	округление до ближайшего большего целого числа
cos	косинус
random	выводит случайное число от 0 до аргумента функции.
exp	вычисление экспоненты
fabs	абсолютная величина (числа с плавающей точкой)
floor	округление до ближайшего меньшего целого числа

3.3. Математические функции

fmod	вычисление остатка от деления нацело для чисел с плавающей точкой
frexp	разбивает число с плавающей точкой на мантиссу и показатель степени.
ldexp	умножение числа с плавающей точкой на целую степень двух
log	натуральный логарифм
log10	логарифм по основанию 10
modf(x,p)	извлекает целую и дробную части (с учетом знака) из числа с плавающей точкой
pow(x,y)	результат возведения x в степень y, xy
sin	синус
sinh	гиперболический синус
sqrt	квадратный корень
tan	тангенс
tanh	гиперболический тангенс

3.4. Линейный алгоритм

Линейный алгоритм – тип алгоритма, в котором действия выполняются однократно в заданном порядке

3.4. Линейный алгоритм

Пример. Вычисление площади и периметра квадрата по известной длине.

```
#include "stdafx.h"
#include <iostream>
#include <math.h>
using namespace std;
int _tmain()
{
  float dl, s;
  cout << "Enter the length of side:"; // Вывод литеральной строки
  cin >> dl; // Ввод
  s = pow(dl, 2); // Вызов функции pow
  cout << "s = " << s << endl; // Вывод строки "s =" , значения s, кон. стр.
  cout << "p = " << dl * 4;
  return 0;
}
```

3.4. Линейный алгоритм

Задания.

1. Нарисовать прямоугольник из звёздочек в консоли.
2. Вычислить по известному радиусу площадь круга и длину окружности.
3. Пользователь вводит два числа. Оба числа не являются нулём. Вывести их сумму, разность, произведение, частное.
4. Пользователь вводит число. Вывести на экран квадрат, куб и четвёртую степень этого числа.
5. Пользователь вводит количество секунд, прошедшее с некоторого момента. Вывести на экран это количество времени в формате: дни часы минуты секунды.
6. Банкомат имеет купюры достоинством 5000, 1000, 500, 100 руб. Пользователь банкомата вводит сумму, кратную 100 руб. Пользователь банкомата должен получить минимальное количество купюр. Подсказка: использовать целочисленное деление и получение остатка от целочисленного деления.