C++ course

name: Виктория Александровна
e-mail: kewtree1408@gmail.com

source: https://bitbucket.org/Kewtree/coursecpp/src
login: coursecpp
password: 12345

General Plans

1. Inception
2. Statements
3. Functions
4. Pointers
5. Massives
6. Structures
7. Files
8. OOP: encapsulation
9. OOP: inheritance
10. OOP: polimorph
11. Test
12. Game over

Statements

1. Repeat (type-enum, switch-case)
2. while
3. do-while
4. for
5. functions
6. function-definitions

Hometask: switch-case, enum

Пусть у нас есть файл известного размера. И есть CD, DVD диски, Flash 4Gb, 8Gb. По введенному размеру файла и установленным физ. носителям определить, уместиться ли наш файл или нет. Все данные о файле будем вводить в Gb.

Statement: while

```
while (условие) {
    операторы;
    изменение счетчика;
}
```

Какого типа <условие>?
Как изменяется счетчик?
Какое соответсвие между услвоием и счетчиком?

Examples: while

1. `table_1_2_3.cpp`

Ввести число, в соответсвии с этим числом вывести таблицу из этих чисел, их квадратов и кубов.

2. `fibonacci.cpp`

Вывести последовательность чисел Фиббоначи.

3. `sqrt.cpp`

По введенному числу выдаем через табуляцию значение квадратного корня из этого числа

Statement: do-while

```
do {
   операторы;
   изменение счетчика;
} while (условие);
```

Какого типа <условие>?
Как изменяется счетчик?
Какое соответсвие между услвоием и счетчиком?

Example: do-while

`guess_random.cpp`

Программа сгенерирует некоторое случайное число, а мы попытаемся его угадать.

Как сделать так, чтоб мы никогда не догадались, какое число задумано программой?

Statement: for

```
for (expr1; expr2; expr3) {
   operators;
}
```

expr1 - служит для инициализации какой-либо переменной, выполняющей роль счетчика итераций

expr2 - используется как проверочное условие на выход из цикла, выход из цикла -- если ложь

expr3 - служит, как правило, для приращения счетчика цикла либо содержит действия, влияющие на проверочное условие expr2

Statement: for

```
for (expr1; expr2; expr3) {
   operators;
}
```

эквивалентно:

```
expr1;
while(expr2) {
   operators;
   expr3;
}
```

Statement: for

```
for (expr1; expr2; expr3) {
   operators;
}
```

Каждое из 3х выражений (expr1,expr2,expr3) не обязательно должны присутствовать, но ';' - обязана быть.

```
for (;;)
    cout << "Infinity\n";
```

Examples: for

`for_example.cpp`

Пример, выполняющий сложение чисел от 0 до 10 с разными вариантами цикла for.

`fibo.cpp`

Вычисление числа Фиббоначчи с помощью цикла for.

Functions

Функция (в программировании) — это проименованная часть программы, которая может вызываться из других частей программы столько раз, сколько необходимо.

Функция (в математике)— это «закон», по которому каждому элементу одного множества (называемому ***областью определения***) ставится в соответствие некоторый элемент другого множества (называемого ***областью значений***).

Функция (в философии) — обязанность, круг деятельности.

Functions

```
int main() {
   ...
   return 0;
}

void print_hello() {
   cout << "Hello, world\n";
}

float min(float a, float b){
   return ((a>b)?a:b);
}
```

Functions

<возвращаемый тип> <название функции> (список входных параметров через ',') {
    return <возвращаемое значение>;
}

Возвращаемое значение обязательно совпадает с возвращаемым типом.

Functions: definitions

Основные определения:

1. Объявление функции (прототип, сигнатура) - сообщение компилятору ее имени, типа входных параметров и возвращаемого значения.

```
int main();
void print_hello();
float min(float, float);
```

2. Определение функции (реализация) - объявление вместе с телом функции

```
int main() { return 0; }
void print_hello() { cout << "Hello, world\n"; }
float min(float a, float b){ return ((a>b)?a:b); }
```

Functions

Существует три способа объявления функций:

1. Разместить прототип функции в заголовочный файл (*.h), а затем включить этот файл в исходный текст программы через директиву #include.
2. Записать прототип функции в тот файл, где она используется.
3. Определить функцию до того, как она будет впервые вызвана.

Example: function

`even-odd.cpp`

Пример функций, которые проверяют число на четность и нечетность.

Hometask

1. В бесконечном цикле по введенному числу выдавать синус и косинус этого числа (в градусах).
2. Проверка, является ли введенное число простым или нет.
3. Программа НОД -- наибольший общий делитель для 2х чисел. По алгоритму Евклида.

4.* Нахождение суммы бесконечного ряда с заданной точностью.

$$e^x = 1 + x + \frac{x^2}{2!} + \ldots + \frac{x^n}{n!} + o(x^n)$$