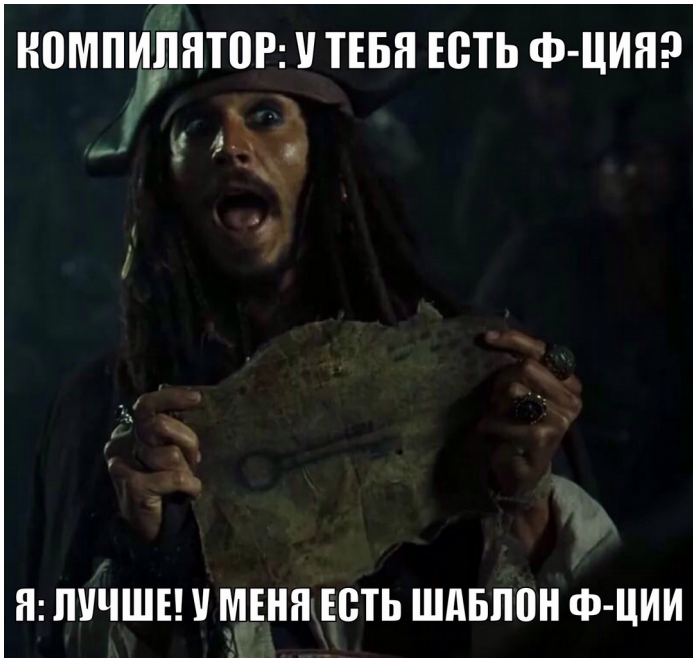
0. Сигнатура ф-ции – список её аргументов.
1. Полиморфизм – способность иметь множество форм.
2. Полиморфизм функций (перегрузка функции) – свойство языка C++, позволяющее привязать к одному имени более чем одну ф-цию. Эту привязку возможно осуществить за счёт различия в количестве аргументов у двух разных функций с одинаковым именем.
3. Вот так работают аргументы по умолчанию:

```
#include <cstdlib>
#include <iostream>

void func(int a, int c = 10)
{
      std::cout << "a = " << a << "\n";
            std::cout << "c = " << c << "\n";
   return;
}

int main()
{
   int a = 0;
   int c = 0;
   func(a); //Выводит 0 и 10
   func(a, c);//Выводит 0 и 0
   return 0;
}
```

4. Шаблон ф-ции — обобщённое описание ф-ции, в котором тип является параметром. Передавая шаблону тип в качесве параметра, можно заставить компилятор сгенерировать ф-цию для этого конкретного типа. Шаблон нужен, чтобы не прописывать вручную одну и ту же ф-цию только для разных типов данных. Шаблон отличается от ф-ции примерно так же, как рисунок ключа отличается от самого ключа. Ограничения логичны: шаблон может быть изначально написан, исходя из предположения, что он будет применен к определённому набору типов. Например, в теле шаблона имеется операция сравнения, а его применяют к структуре.



5. Явная специализация — задание функции специальным образом для определенного типа. Если ф-ция для определённой сигнатуры описана отдельно, то для этой сигнатуры компилятор не будет создавать новую ф-цию по шаблону, а воспользуется отдельным описанием.
   Пример явной специализации:
   template <> void function<type> (type &a, type&b)

{
...
}

6. Ссылка в C++ представляет собой имя, которое является альтернативным именем для ранее объявленной пременной.
   Чтобы b стало альтернотивным именнем для a пишем:
   int a;
   int& b = a; //& - это не взятие адреса в данном случае.
   Нельзя сначала объявить ссылку, а потом присвоить ей значение.
   Нельзя присвоить ссылке значение два раза.
   Можно осуществлять передачу аргументов в ф-цию по ссылке, она позволяет в вызываемой ф-ции получить доступ к переменным вызывающей.
7. При использовании ключевого слова inline в процессе компиляции вместо вызова ф-ции подставляется её код (такая ф-ция называется встроенной). Это позволяет ускорить выполнение программы. Если аргумент такой ф-ции — это выражение, то ф-ции передаётся значение выражения. Для создания встроенной ф-ции необходимо перед оъбявлением встроенной ф-ции написать inline. Н-р:
   inline void func(...){...};
8. Примеры использования auto:
   ...
   auto a = 0.0; // создали пременную a типа double.
   Int b = 0;
   auto c = b; // создали переменную c типа int.
   ...

   Пример использования decltype:
   ...
   decltype(a * b) c = (a*b); // создали пременную с того же типа, что и a*b
   ...

decltype может понадобиться в том случае, если мы не знаем заранее, какой тип будет иметь a*b. Н-р в случее если a,b — пременные структуры описывающей матрицу, а * переопределено, как операция соответствующая матричному умножению.