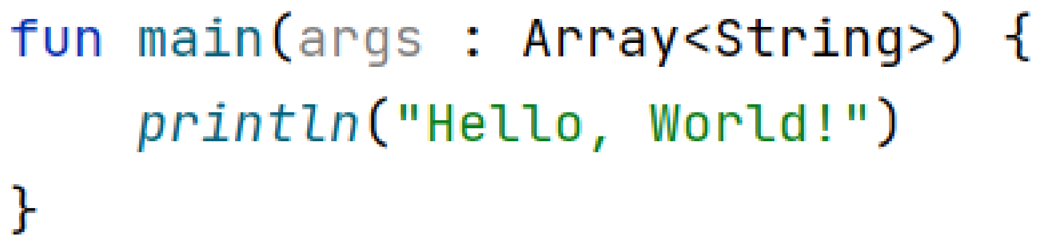
Agenda

- Hello world example
- Compilation
- Bytecode

কটলিন একটি ওপেন সোর্স, স্ট্যাটিকালি-টাইপড, মাল্টি প্যারাডাইম প্রোগ্রামিং ল্যাঙ্গুয়েজ, যা অবজেক্ট-ওরিয়েন্টেড প্রোগ্রামিং ও ফাংশনাল প্রোগ্রামিং উভয়ই সাপোর্ট করে। কটলিন প্রোগ্রামিং ল্যাঙ্গুয়েজের সিনট্যাক্স সি, সি++, সি#, জাভা প্রোগ্রামিং ল্যাঙ্গুয়েজের মত সিনট্যাক্স। আর কটলিনে - লাইনের শেষে সেমিকোলন দেয়া অপশনাল, না দিলেও কাজ করবে।

Hello Worl Program

REC



```
fun main(args : Array<String>) {
    println("Hello, World!")
}
```

কটলিন প্রগ্রাম রান করে এক্সিকিউটেবল কটলিন বাইট কোড তৈরি করতে JVM এর সাহায্য নেয়। আর কটলিন প্রগ্রামের এন্ট্রি পয়েন্ট হচ্ছে main ফাংকশন। অর্থাৎ কটলিন প্রগ্রাম রান করে main ফাংকশন থেকে।
আর println একটি সিস্টেম ফাংশন। কোনো কিছু কন্সোলে প্রিন্ট করার জন্য print ও println ফাংশন ব্যবহার করা হয়।

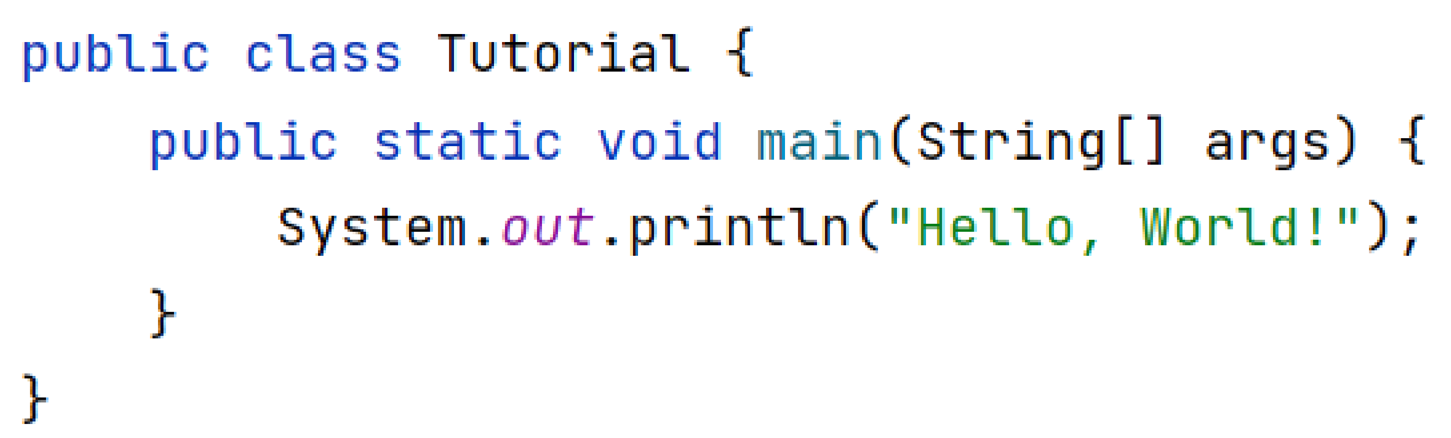
এখানে println একটি সিস্টেম ফাংশন। তাহলে ফাংকশন কি? ফাংকশন নিয়ে বিস্তারিত ধারণা পরে দেয়া হবে। এখানে শুধুমাত্র বোঝার সুবিধার জন্য হালকা আলোচনা করা হলো।

Function is a set of instructions to perform specific task.

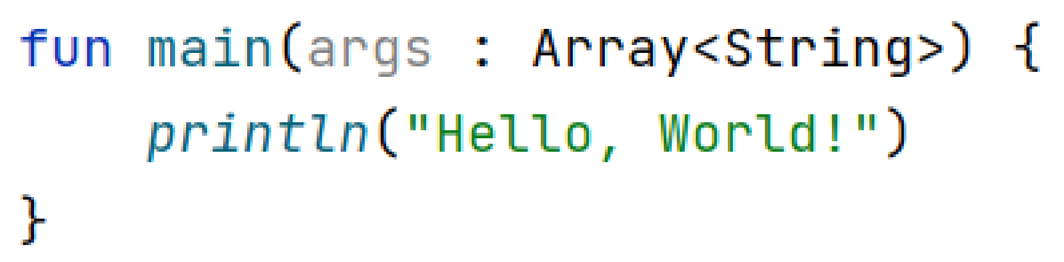
Inputs → Function → Output

```
fun name(args arguments) {
   // Code
}
```

ফাংশনকে নেম ব্লকও বলা হয়।
এখন দেখি প্রগ্রামটি জাভাতে কিভাবে লেখা হতো।



```
public class Tutorial {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
```



```
fun main(args : Array<String>) {
    println("Hello, World!")
}
```

পাবলিক ক্লাস, ক্লাস নেম এর ভিতরে পাবলিক স্ট্যাটিক main ফাংশন বা মেথড। যদি কোন জাভা কোডকে এক্সিকিউট করতে হলে ক্লাস বা ক্লাস ফাইলের প্রযোজন হয়। অর্থাৎ জাভা ভার্চুয়াল মেশিন বা JVM শুধুমাত্র ক্লাস ফাইল নিয়েই কাজ করে। ঠিক একইভাবে কটলিন ও জাভা ভার্চুয়াল মেশিন বা JVM এ রান করে। অর্থাৎ কটলিন রানটাইমে কোড এক্সিকিউট করার জন্য JVM এর প্রযোজন হয়। কিন্তু আমরা জানি কটলিন main ফাংশন কোন ক্লাসের ভিতর থাকে না। বা কোন ক্লাসের প্রয়োজন হয় না। এখন প্রশ্ন আসে কটলিন কি তাহলে রান টাইমে ইন্টার্নালি ক্লাস তৈরি কর? এর উত্তর হ্যা।

কটলিন কোড যখন এক্সিকিউট হয়, তখন রানটাইমে কটলিন কম্পাইলার ইন্টার্নালি একটি ক্লাস ফাইল তৈরি করে মেমোরিতে লোড করে। সতরাং কটলিন রানটাইমে জাভার মতোই কাজ করে বা জাভার মতোই আচরণ করে।