## **Assignment 1, Mobile Programming**

Bolatova Aruzhan(23MD0393)

# **Exercise 1: Kotlin Syntax Basics**

**Variables and Data Types:** 

```
    □ aru_task1.kt ×

            fun main() {
                val age: Int = 22
                val doubb: Double = 3.14
                val name: String = "ARUZHAN!"
nAc
                val bl: Boolean = true
kam
mpl
                println("Integer: $age")
                println("Double: $doubb")
am
                println("String: $name")
sk1.k
                println("Boolean: $bl")
  Aru_task1Kt ×
```

```
Integer: 22
Double: 3.14
String: ARUZHAN!
Boolean: true
```

#### **Conditional Statements:**

```
5 is positive
```

### Loops:

```
Image: State of the content of
```

### **Collections:**

```
fun collectnum() {
    val numbers = listOf(1, 2, 3, 4, 5)
    var sum = 0
    for (number in numbers) {
        sum += number
    }
    println("Sum: $sum")
}
```

Sum: 15

## **Exercise 2: Kotlin OOP (Object-Oriented Programming)**

```
aru_task2.kt ×
                                                                                           1 ▶ fun main() {
                                                                                          ▲1 ▲1 ≪2 ^
          val person = Person( name: "Aruzhan Bolatova", age: 22, email: "aru.b@gmail.com")
          person.displayInfo()
          val employee = Employee( name: "Jay Park", age: 24, email: "jay.park@gmail.com", salary: 50000.0)
          employee.displayInfo()
          val account = BankAccount( balance: 1000.0)
          account.deposit( amount: 500.0)
          account.withdraw( amount: 300.0)
          account.withdraw( amount: 1500.0) // Should not allow
open class Person(val name: String, val age: Int, val email: String) {
              println("Name: $name, Age: $age, Email: $email")
       class Employee(name: String, age: Int, email: String, val <u>salary</u>: Double) : Person(name, age, email
       class BankAccount(private var balance: Double) {
```

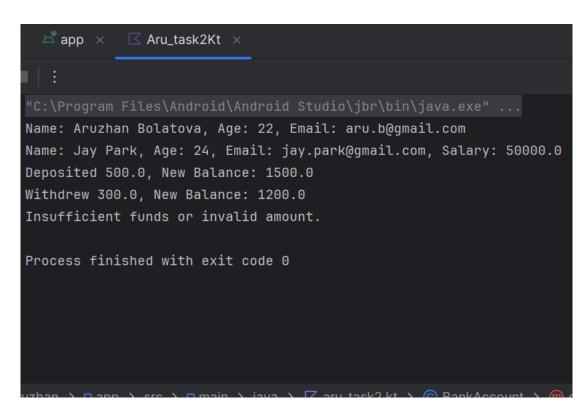
```
class BankAccount(private var balance: Double) {

fun deposit(amount: Double) {

   if (amount > 0) {
      balance += amount
      println("Deposited $amount, New Balance: $balance")
   }

fun withdraw(amount: Double) {
   if (amount > 0 && balance >= amount) {
      balance -= amount
      println("Withdrew $amount, New Balance: $balance")
   } else {
      println("Insufficient funds or invalid amount.")
   }

fun getBalance(): Double {
   return balance
}
```



#### **Exercise 3: Kotlin Functions**