Міністерство освіти і науки України  
Національний технічний університет України

«Київський політехнічний інститут ім. І. Сікорського»

Кафедра інженерії програмного забезпечення в енергетиці

Лабораторна робота № 6

з курсу: «*Розробка програмного*

*забезпечення мобільних пристроїв*»

**Виконав:**  
студент 4-го курсу,  
групи ТВ-13  
Романін Анатолій Олександрович

Посилання на GitHub репозиторій:https://github.com/JIAIM/mobile\_software

**Перевірив:**

Недашківський О.Л.

Київ 2024/2025

Лабораторна робота № 6

### Теоретичні Відомості

#### Основні поняття та формули

1. **Номінальна потужність (Pn):** Величина, яка характеризує здатність електричної машини виконувати роботу при номінальних умовах. Вимірюється в кіловатах (кВт).
2. **Коефіцієнт використання потужності (Kv):** Показник ефективності використання потужності. Визначається як відношення фактичного навантаження до номінального. Значення Kv знаходиться в межаx 0≤Kv≤1.
3. **Коефіцієнт реактивної потужності (tgφ):** Відображає співвідношення між активною та реактивною складовими повної потужності. Реактивна потужність Q обчислюється за формулою: Q=P⋅tgφ, де P — активна потужність.
4. **Струм у ланцюзі (I):** Струм обчислюється як:

I=P3⋅U⋅cosφ⋅η, де:

* + U — напруга;
  + cosφ — коефіцієнт потужності;
  + η — ККД (коефіцієнт корисної дії).

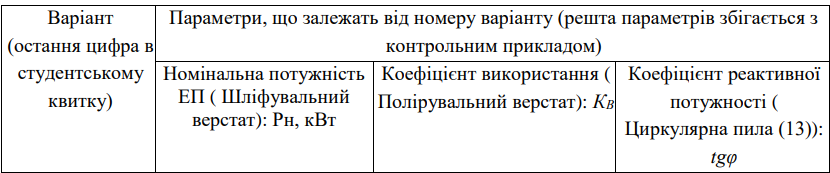
1. **Активна, реактивна та повна потужності:**
   * P — активна потужність;
   * Q — реактивна потужність;
   * S — повна потужність: S=.
2. **Квадратурний метод:** Для розрахунків загальної потужності групи машин використовується метод сумування квадратів номінальних потужностей:

Pзаг=

**Завдання :**

Створіть мобільний калькулятор для розрахунку електричних навантажень об’єктів з використанням методу впорядкованих діаграм. Цехова мережа складається з трьох типових цехів які під’єднується до трьох різних розподільчих шин (ШР1-ШР3) та кількох крупних електроприймачів (ЕП). Для спрощення приймемо що склад, номенклатура і характеристики ЕП всіх трьох цехів однакові. На основі складу ЕП та їх характеристик необхідно розрахувати силове навантаження цехової мережі

**Варіант 8 :**

**Хід виконання:**

Для обчислення формул, також для більш зручного виводу створена функція round, що округлює значення до сотих.

fun round(x: Double): Double {  
 return BigDecimal(x).setScale(4, RoundingMode.*HALF\_UP*).toDouble()  
}

Початкові дані для обрахунків беремо із таблиці 6.6.

val Un = 0.38  
val n\_data = *intArrayOf*(4, 2, 4, 1, 1, 1, 2, 1)  
val Pn\_data = *intArrayOf*(Pn, 14, 42, 36, 20, 40, 32, 20)  
val Kv\_data = *doubleArrayOf*(0.15, 0.12, 0.15, 0.3, 0.5, Kv, 0.2, 0.65)  
val tgphi\_data = *doubleArrayOf*(1.33, 1.0, 1.33, tgphi, 0.75, 1.0, 1.0, 0.75)  
val n\_Pn\_data = *mutableListOf*<Int>()  
val n\_Pn\_Kv\_data = *mutableListOf*<Double>()  
val n\_Pn\_Kv\_tgphi\_data = *mutableListOf*<Double>()  
val n\_Pn\_Pn\_data = *mutableListOf*<Int>()  
val Ip\_data = *mutableListOf*<Double>()

Далі згідно до таблиці 6.7 необхідно обрахувати наступні значення:

for (i in n\_data.*indices*) {  
 val n = n\_data[i]  
 val Pn = Pn\_data[i]  
 val Kv = Kv\_data[i]  
 val tgphi = tgphi\_data[i]  
 n\_Pn\_data.add(n \* Pn)  
 n\_Pn\_Kv\_data.add(*round*(n\_Pn\_data[i] \* Kv))  
 n\_Pn\_Kv\_tgphi\_data.add(*round*(n\_Pn\_Kv\_data[i] \* tgphi))  
 n\_Pn\_Pn\_data.add(n \* Pn \* Pn)  
 Ip\_data.add(*estimatedCurrents1level*(n\_Pn\_data[i]))  
}  
  
val SHR1\_n = n\_data.*sum*()  
val SHR1\_n\_Pn = n\_Pn\_data.*sum*()  
val SHR1\_Kv = BigDecimal(n\_Pn\_Kv\_data.*sum*()/n\_Pn\_data.*sum*()).setScale(2, RoundingMode.*HALF\_UP*)  
val SHR1\_n\_Pn\_Kv = n\_Pn\_Kv\_data.*sum*()  
val SHR1\_n\_Pn\_Kv\_tgphi = *round*(n\_Pn\_Kv\_tgphi\_data.*sum*())  
val SHR1\_n\_Pn\_Pn = n\_Pn\_Pn\_data.*sum*()  
val SHR1\_ne = SHR1\_n\_Pn\*SHR1\_n\_Pn/SHR1\_n\_Pn\_Pn + 1  
val SHR1\_Kp = 1.25 //data from table 6.3  
val SHR1\_Pp = *round*(SHR1\_Kp\*SHR1\_n\_Pn\_Kv)  
val SHR1\_Qp = 1.0\*SHR1\_n\_Pn\_Kv\_tgphi  
val SHR1\_Sp = *round*(*sqrt*(SHR1\_Pp\*SHR1\_Pp+SHR1\_Qp\*SHR1\_Qp))  
val SHR1\_Ip = *round*(SHR1\_Pp/Un)  
  
val transfor\_n = 2  
val transfor\_Pn = 100  
val transfor\_n\_Pn = transfor\_n\*transfor\_Pn  
val transfor\_Kv = 0.2  
val transfor\_tgphi = 3.0  
val transfor\_n\_Pn\_Kv = transfor\_n\_Pn\*transfor\_Kv  
val transfor\_n\_Pn\_Kv\_tgphi = transfor\_n\_Pn\_Kv\*transfor\_tgphi  
val transfor\_n\_Pn\_Pn = transfor\_n\_Pn\*transfor\_Pn  
val transfor\_Ip = *estimatedCurrents1level*(transfor\_n\_Pn)  
  
val sushi\_n = 2  
val sushi\_Pn = 120  
val sushi\_n\_Pn = sushi\_n\*sushi\_Pn  
val sushi\_Kv = 0.8  
val sushi\_tgphi = ' '  
val sushi\_n\_Pn\_Kv = sushi\_n\_Pn\*sushi\_Kv  
val sushi\_n\_Pn\_Kv\_tgphi = ' '  
val sushi\_n\_Pn\_Pn = sushi\_n\_Pn\*sushi\_Pn  
val sushi\_Ip = *estimatedCurrents1level*(sushi\_n\_Pn)  
  
val all\_n = 81  
val all\_n\_Pn = 2330  
val all\_n\_Pn\_Kv = 752  
val all\_Kv = BigDecimal(all\_n\_Pn\_Kv.toDouble()/all\_n\_Pn).setScale(2, RoundingMode.*HALF\_UP*)  
val all\_n\_Pn\_Kv\_tgphi = 657  
val all\_n\_Pn\_Pn = 96399  
val all\_ne = all\_n\_Pn\*all\_n\_Pn/all\_n\_Pn\_Pn  
val all\_Kp = 0.7 //from table  
val all\_Pp = *round*(all\_Kp\*all\_n\_Pn\_Kv)  
val all\_Qp = all\_Kp\*all\_n\_Pn\_Kv\_tgphi  
val all\_Sp = *round*(*sqrt*(all\_Pp\*all\_Pp+all\_Qp\*all\_Qp))  
val all\_Ip = *round*(all\_Pp/Un)

Для відображення цих даних створимо відповідно таблицю в яку будемо додавати дані для виводу:

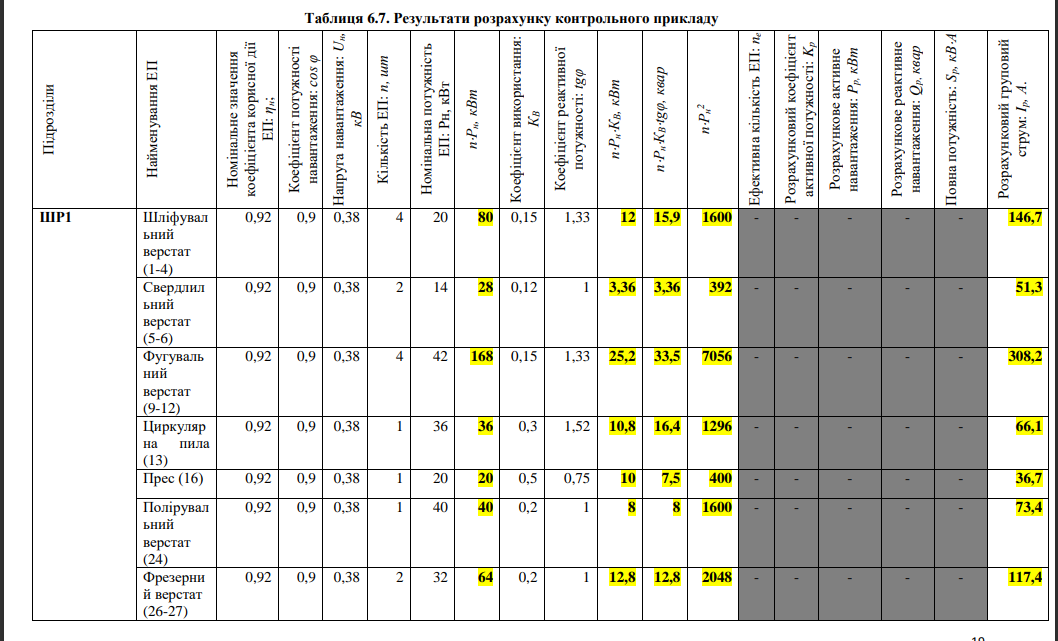
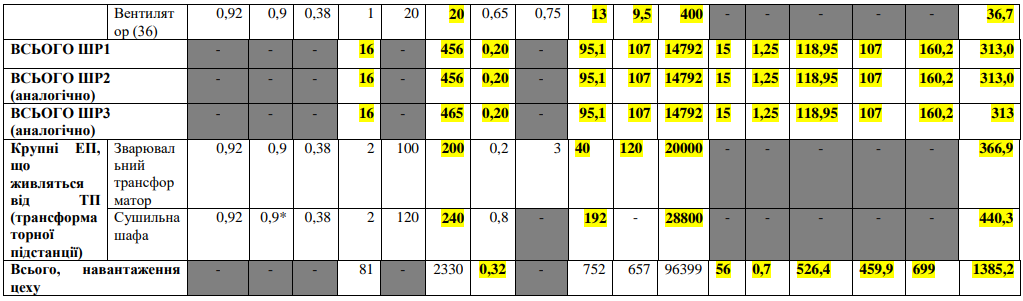
val table = *mutableListOf*<List<String>>()  
for (i in n\_data.*indices*) {  
 table.add(  
 *listOf*(  
 (i+1).toString(),  
 String.*format*("%.2f", 0.92),  
 String.*format*("%.1f", 0.9),  
 String.*format*("%.2f", Un),  
 n\_data[i].toString(),  
 Pn\_data[i].toString(),  
 n\_Pn\_data[i].toString(),  
 String.*format*("%.2f", Kv\_data[i]),  
 String.*format*("%.2f", tgphi\_data[i]),  
 String.*format*("%.2f", n\_Pn\_Kv\_data[i]),  
 String.*format*("%.3f", n\_Pn\_Kv\_tgphi\_data[i]),  
 n\_Pn\_Pn\_data[i].toString(),  
 "", "", "", "", "", String.*format*("%.4f", Ip\_data[i])  
 )  
 )  
}  
  
for (i in 1..3){  
 table.add(  
 *listOf*(  
 "SHR"+i.toString(),  
 "",  
 "",  
 "",  
 SHR1\_n.toString(),  
 "",  
 SHR1\_n\_Pn.toString(),  
 String.*format*("%.2f", SHR1\_Kv),  
 "",  
 String.*format*("%.2f", SHR1\_n\_Pn\_Kv),  
 String.*format*("%.3f", SHR1\_n\_Pn\_Kv\_tgphi),  
 SHR1\_n\_Pn\_Pn.toString(),  
 SHR1\_ne.toString(),  
 String.*format*("%.2f", SHR1\_Kp),  
 String.*format*("%.2f", SHR1\_Pp),  
 String.*format*("%.2f", SHR1\_Qp),  
 String.*format*("%.2f", SHR1\_Sp),  
 String.*format*("%.4f", SHR1\_Ip),  
 )  
 )  
}  
  
table.add(  
 *listOf*(  
 "Trans",  
 String.*format*("%.2f", 0.92),  
 String.*format*("%.1f", 0.9),  
 String.*format*("%.2f", Un),  
 transfor\_n.toString(),  
 transfor\_Pn.toString(),  
 transfor\_n\_Pn.toString(),  
 String.*format*("%.2f", transfor\_Kv),  
 String.*format*("%.2f", transfor\_tgphi),  
 String.*format*("%.2f", transfor\_n\_Pn\_Kv),  
 String.*format*("%.3f", transfor\_n\_Pn\_Kv\_tgphi),  
 transfor\_n\_Pn\_Pn.toString(),  
 "", "", "", "", "", String.*format*("%.4f", transfor\_Ip)  
 )  
)  
  
table.add(  
 *listOf*(  
 "Shafa",  
 String.*format*("%.2f", 0.92),  
 String.*format*("%.1f", 0.9),  
 String.*format*("%.2f", Un),  
 sushi\_n.toString(),  
 sushi\_Pn.toString(),  
 sushi\_n\_Pn.toString(),  
 String.*format*("%.2f", sushi\_Kv),  
 "",  
 String.*format*("%.2f", sushi\_n\_Pn\_Kv),  
 "",  
 transfor\_n\_Pn\_Pn.toString(),  
 "", "", "", "", "", String.*format*("%.4f", sushi\_Ip)  
 )  
)  
  
table.add(  
 *listOf*(  
 "All",  
 "",  
 "",  
 "",  
 all\_n.toString(),  
 "",  
 all\_n\_Pn.toString(),  
 String.*format*("%.2f", all\_Kv),  
 "",  
 all\_n\_Pn\_Kv.toString(),  
 all\_n\_Pn\_Kv\_tgphi.toString(),  
 all\_n\_Pn\_Pn.toString(),  
 all\_ne.toString(),  
 String.*format*("%.2f", all\_Kp),  
 String.*format*("%.2f", all\_Pp),  
 String.*format*("%.2f", all\_Qp),  
 String.*format*("%.2f", all\_Sp),  
 String.*format*("%.4f", all\_Ip),  
 )  
)

Для взаємодії з користувачем створимо 3 поля для вводу даних та кнопку для обрахунку задачі:

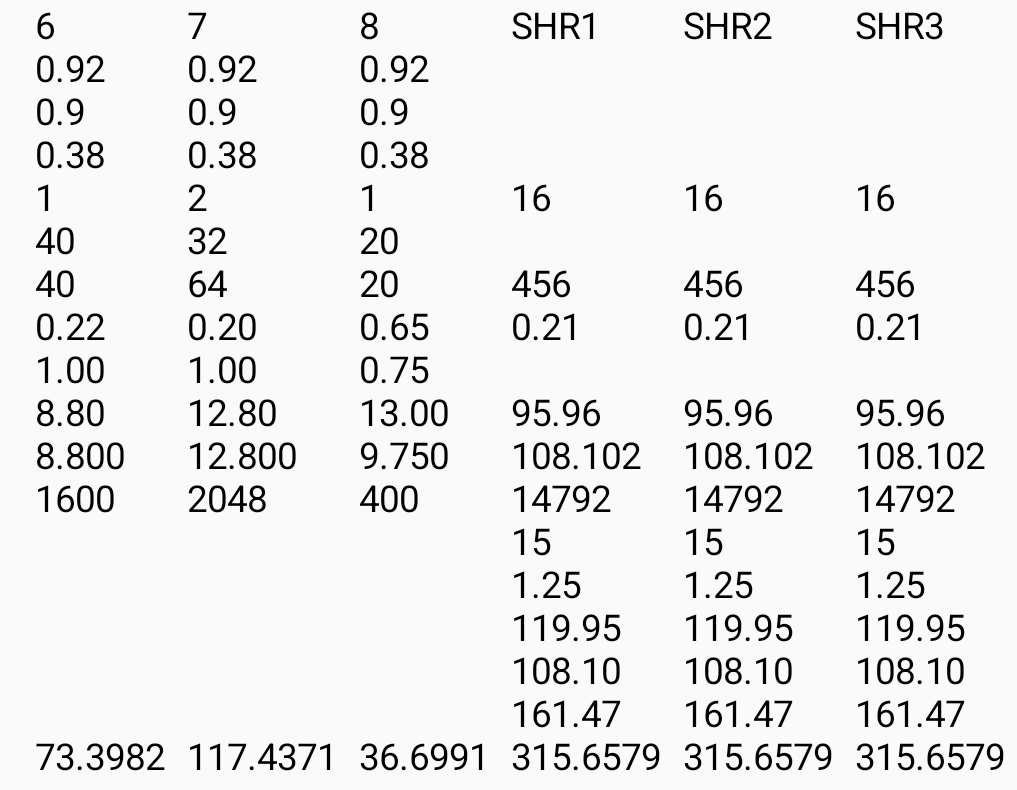
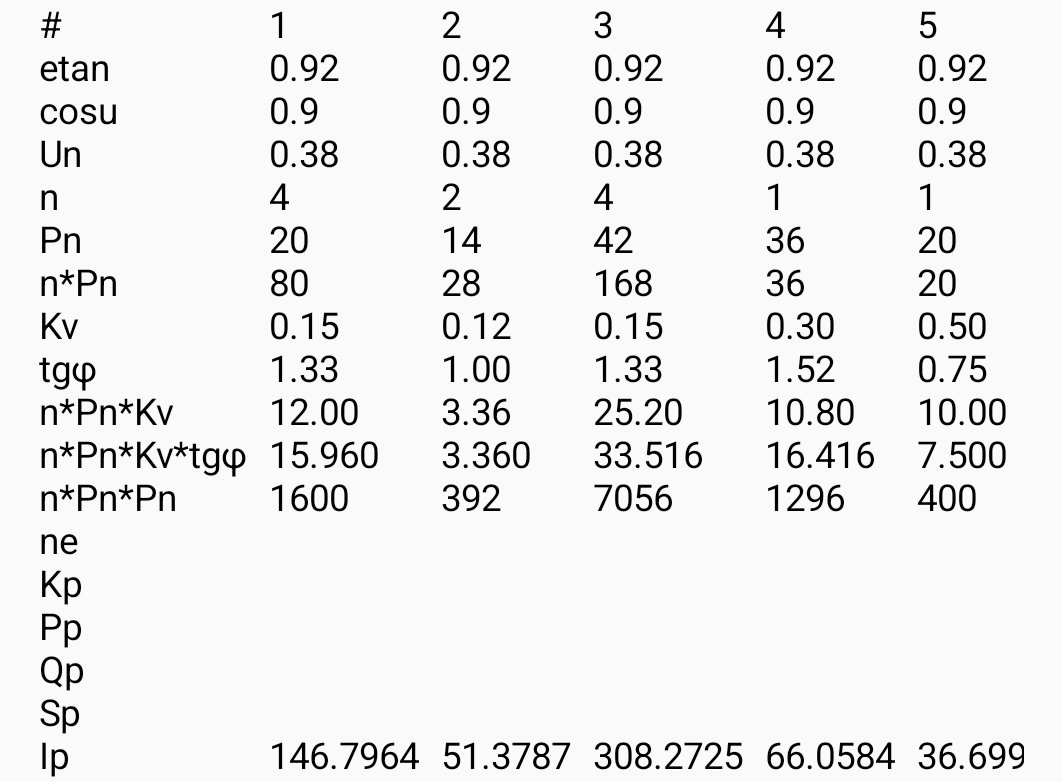
@Composable  
fun CalculatorApp() {  
 var Pn by remember **{** *mutableStateOf*("20") **}** var Kv by remember **{** *mutableStateOf*("0.22") **}** var tgphi by remember **{** *mutableStateOf*("1.52") **}** var tableData by remember **{** *mutableStateOf*(*listOf*<List<String>>()) **}** Column(  
 modifier = Modifier  
 .*fillMaxSize*()  
 .*padding*(16.*dp*)  
 ) **{** OutlinedTextField(  
 value = Pn,  
 onValueChange = **{** Pn = **it }**,  
 label = **{** Text("Pn") **}**,  
 modifier = Modifier.*fillMaxWidth*().*padding*(bottom = 8.*dp*)  
 )  
  
 OutlinedTextField(  
 value = Kv,  
 onValueChange = **{** Kv = **it }**,  
 label = **{** Text("Kv") **}**,  
 modifier = Modifier.*fillMaxWidth*().*padding*(bottom = 8.*dp*)  
 )  
  
 OutlinedTextField(  
 value = tgphi,  
 onValueChange = **{** tgphi = **it }**,  
 label = **{** Text("tgφ") **}**,  
 modifier = Modifier.*fillMaxWidth*().*padding*(bottom = 16.*dp*)  
 )  
  
 Button(  
 onClick = **{** tableData = *calculateTable*(Pn.*toInt*(), Kv.*toDouble*(), tgphi.*toDouble*())  
 **}**,  
 modifier = Modifier.*fillMaxWidth*()  
 ) **{** Text("Рассчитать")  
 **}** Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 if (tableData.*isNotEmpty*()) {  
 Row(modifier = Modifier.*horizontalScroll*(rememberScrollState())) **{** Column(  
 modifier = Modifier.*padding*(4.*dp*)  
 ) **{** Text("#")  
 Text("etan")  
 Text("cosu")  
 Text("Un")  
 Text("n")  
 Text("Pn")  
 Text("n\*Pn")  
 Text("Kv")  
 Text("tgφ")  
 Text("n\*Pn\*Kv")  
 Text("n\*Pn\*Kv\*tgφ")  
 Text("n\*Pn\*Pn")  
 Text("ne")  
 Text("Kp")  
 Text("Pp")  
 Text("Qp")  
 Text("Sp")  
 Text("Ip")  
 **}** tableData.*forEach* **{** row **->** Column(modifier = Modifier.*padding*(4.*dp*)) **{** row.*forEach* **{** cell **->** Text(cell)  
 **}  
 }  
 }  
 }** }  
 **}**}

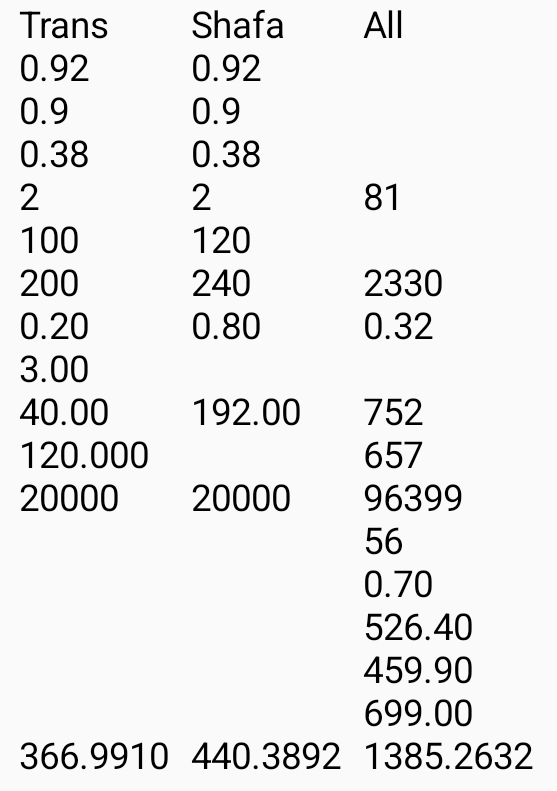
**Перевірка:**

Для перевірки результатів введемо початкові дані із контрольного прикладу:

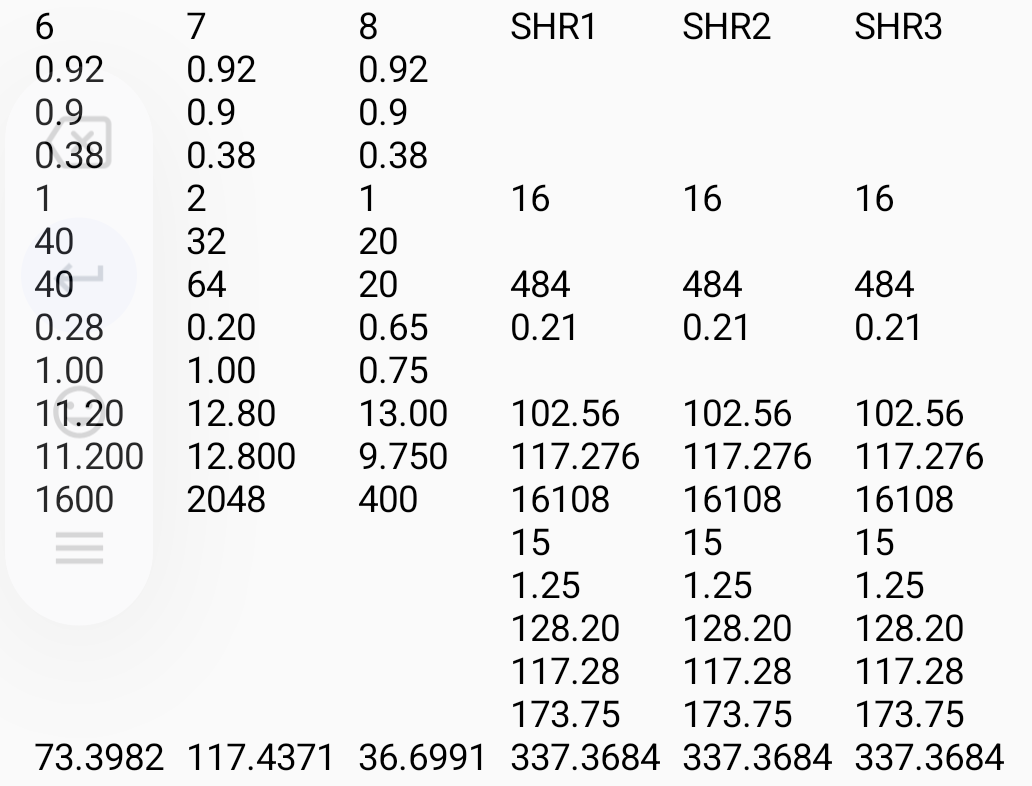
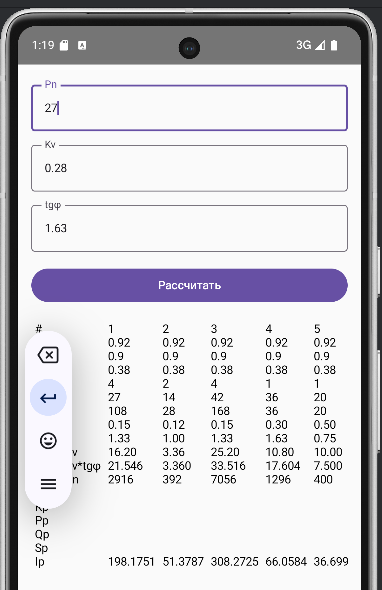
 

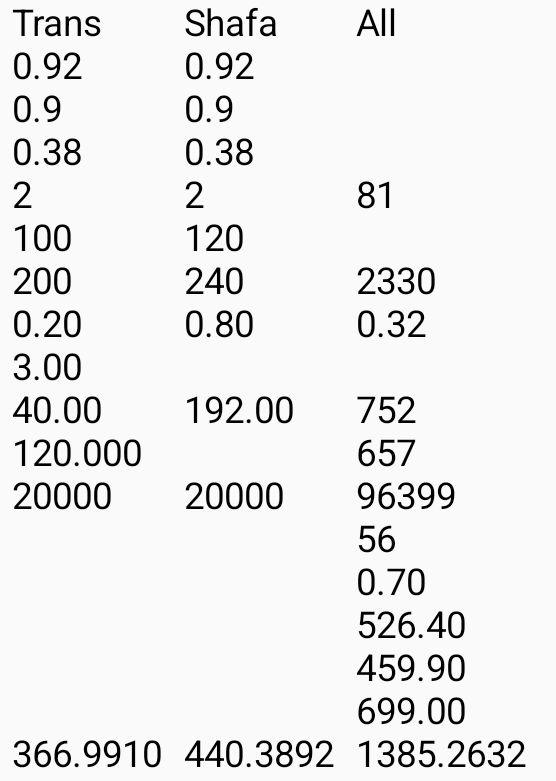






Результати аналогічні, тепер напишимо дані для варіанту 8





**Висновок**

У результаті виконання практичної роботи було розроблено мобільний додаток для розрахунку основних параметрів електричних машин, зокрема номінальної потужності, коефіцієнта використання та реактивної потужності. Додаток дозволяє користувачам вводити початкові дані, здійснювати обчислення струмів, активної та реактивної потужності, а також отримувати результати у вигляді таблиці у зручному форматі.

Під час роботи ми ознайомилися з основними принципами розрахунку енергетичних параметрів електричних систем, засвоїли методику визначення основних характеристик, таких як активна, реактивна та повна потужність. Ми також опанували навички створення користувацького інтерфейсу з використанням фреймворку Jetpack Compose у середовищі Android Studio, що дозволило автоматизувати розрахунки та візуалізувати результати.