**Прим. \_\_\_**

**ЗАТВЕРДЖУЮ**

Викладач кафедри інженерії програмного забезпечення Національного авіаційного університету

“\_\_” \_\_\_\_\_\_\_\_\_\_ 2021

**Лабораторна робота №1**

**Оцінка вартості розробки програмної системи за моделлю COCOMO**

**на** **8 аркушах**

Студентка групи ПІ-423Б

Факультет кібербезпеки, комп'ютерної та програмної інженерії Національного авіаційного університету

Манжула К.О.

“\_\_\_” \_\_\_\_\_\_\_\_\_\_ 2021

Київ – 2021

1. **Розрахунок за базовим рівнем моделі COCOMO трудовитрат(PM), часу розробки(TM), середньої чисельності(SS) та продуктивності(P).**

Варіант 18 (8): розмір проекту, який розробляється, оцінюється в 85 KLOC.

Для органічного рівню:



Для напівзалежного рівню:



Для вбудованого рівню:



1. **Визначення режиму складності системи за проміжним рівнем моделі COCOMO.**

Варіант 18 (8): розмір проекту, який розробляється, оцінюється в 85 KLOC.

* 1. Значення множників (драйверів) витрат CPLX, SCED змінюються до дуже низьких, АСАР змінюються до низьких, всі інші значення номінальні (взято нормальний рівень).
  2. значення множників (драйверів) витрат: RELY – низький, DATA – дуже високий, CPLX – низький; TIME – дуже високий, STOR – високий, VIRT (PVOL) – дуже високий, TURN (STIME) – дуже високий; ACAP (ASAP) – високий, AEXP – високий, PCAP (PERS) – дуже низький, VEXP (PEXP) – високий, LEXP (LTEX) –високий; MODP (FCIL) – високий, TOOL – дуже високий, SCED – високий.

2.1 Для органічного рівню:



2.1 Для напівзалежного рівню:



2.1 Для вбудованого рівню:



* 1. Для органічного рівню:



* 1. Для напівзалежного рівню:



* 1. Для вбудованого рівню:



1. **COCOMO калькулятор для розрахунку трудоємкості терміну розробки програмного продукту на основі базового рівню вартості.**

Код програми:

Spinner typeSpinner;  
TextView locField;  
HashMap<String, Double> organicCoefficients = new HashMap();  
HashMap<String, Double> semi\_dependentCoefficients = new HashMap<>();  
HashMap<String, Double> builtCoefficients = new HashMap<>();  
ArrayList<String> types = new ArrayList<>();  
  
@Override  
protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 typeSpinner = (Spinner)findViewById(R.id.*ProjectTypeSpinner*);  
 locField = (TextView)findViewById(R.id.*LOK\_Field*);  
 InitCoefficients();  
 InitTypes();  
}  
  
private void InitCoefficients()  
{  
 organicCoefficients.put("ai", 2.4);  
 organicCoefficients.put("bi", 1.05);  
 organicCoefficients.put("ci", 2.5);  
 organicCoefficients.put("di", 0.38);  
  
 semi\_dependentCoefficients.put("ai", 3.0);  
 semi\_dependentCoefficients.put("bi", 1.12);  
 semi\_dependentCoefficients.put("ci", 2.5);  
 semi\_dependentCoefficients.put("di", 0.35);  
  
 builtCoefficients.put("ai", 3.6);  
 builtCoefficients.put("bi", 1.20);  
 builtCoefficients.put("ci", 2.5);  
 builtCoefficients.put("di", 0.32);  
}  
  
private void InitTypes()  
{  
 types.add("Розповсюджений (органічний)");  
 types.add("Напівнезалежний (напіврозподілений)");  
 types.add("Вбудований");  
 InitTypesSpinner();  
}  
  
private void InitTypesSpinner()  
{  
 ArrayAdapter<String> adapter = new ArrayAdapter<String>(this, android.R.layout.*simple\_spinner\_item*, types);  
 adapter.setDropDownViewResource(android.R.layout.*simple\_spinner\_dropdown\_item*);  
 typeSpinner.setAdapter(adapter);  
}  
  
public void CalculateChecker(View view)  
{  
 String check = typeSpinner.getSelectedItem().toString();  
 switch (check)  
 {  
 case "Розповсюджений (органічний)":  
 Calculate(organicCoefficients);  
 break;  
 case "Напівнезалежний (напіврозподілений)":  
 Calculate(semi\_dependentCoefficients);  
 break;  
 case "Вбудований":  
 Calculate(builtCoefficients);  
 break;  
  
 }  
}  
  
private void Calculate(HashMap coefficients)  
{  
 int size = Integer.*parseInt*(locField.getText().toString());  
 double PM, TM, SS, P;  
  
 PM = (Double) coefficients.get("ai") \* Math.*pow*(size, (Double) coefficients.get("bi"));  
 TM = (Double) coefficients.get("ci") \* Math.*pow*(PM, (Double) coefficients.get("di"));  
 SS = PM / TM;  
 P = size / PM;  
  
 String message = "Трудомісткість: " + PM + "\nЧас розробки в місяцях: " + TM +  
 "\nСередня чисельність персоналу: " + SS + "\nПродуктивність: " + P;  
 Toast.*makeText*(getApplicationContext(), message, Toast.*LENGTH\_LONG*).show();  
}

Приклади роботи розробленого калькулятора можна побачити на рисунках 3.1 – 3.3.



Рисунок 3.1 – Калькулятор з незаповненою кількістю рядків

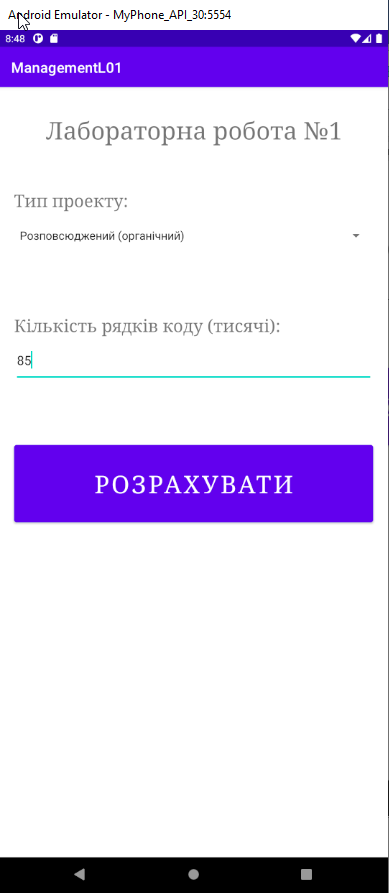


Рисунок 3.2 – Калькулятор із заповненою кількістю рядків

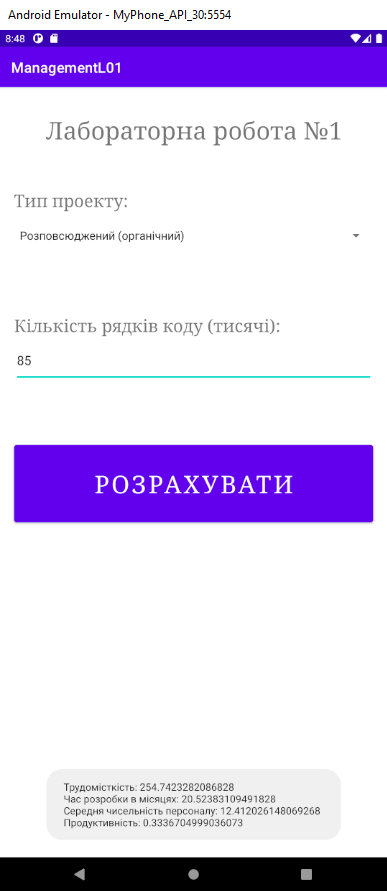


Рисунок 3.3 – Повідомлення користувачеві при натисканні кнопки про розрахунок для обраного типу.