CALCULATOR APPLICATION

មុខវិជ្ជា Mobile App Development
បន្ទប់ ១១៣
ឆ្នាំសិក្សាទី៤ ឆមាសទី២
រៀបរៀងដោយ :

ប៉ែន សម្ផស្ស
សួន អាឡិចសាន់ដឺ
ពន្លឺ អរុណាណោទ័យ

១១ កក្កដា ២០២៤

មាតិកា

១. គោលគំនិតនៃការបង្កើតកម្មវិធី

នៅក្នុង projectមួយនេះ ពួកយើងបានធ្វើការបង្កើតកម្មវិធីគណនាលេខសាមញ្ញមួយដែលជាកម្មវិធីសម្រាប់ប្រើប្រាស់នៅក្នុងទូរស័ព្ទដៃដោយប្រើប្រាស់Flutter។ កម្មវិធីនេះបានអនុញ្ញាតឱ្យអ្នកប្រើប្រាស់ធ្វើការគណនាចំនួនលេខជាលក្ខណៈបែបសាមញ្ញដែលមានដូចជាការគណនាផលបូក ផលដក ផលគុណ និងព្រមទាំងផលចែកផងដែរ។ បន្ថែមពីនេះទៀត កម្មវិធីគណនាលេខនេះក៏បានភ្ជាប់មកជាមួយនូវ interfaceដែលមានភាពស្រស់ស្អាតនិងងាយស្រួលសម្រាប់អ្នកប្រើប្រាស់។

២. មុខងារសំខាន់ៗរបស់ម្មវិធី

១. មុខងារប្រមាណវិធីសម្រាប់គណនា៖ ដែលរួមមាន ប្រមាណវិធីបូក ដក គុណ និងចែក។

២. មុខងារលុបសម្អាត៖ អាចធ្វើការលុបប្រមាណវិធីចាស់ដែលបានគណនារួច និងអាចចាប់ផ្តើមធ្វើការគណនាសារជាថ្មីឡើង។

៣. មុខងារលុបចំនួនខ្ទង់លេខ៖ អាចធ្វើការលុបចំនួនខ្ទង់ខាងក្រោយបំផុតនៃលេខប្រមាណវិធី។

៤. មុខងារផ្លាស់ប្តូរប្រមាណវិធី៖ អាចធ្វើការផ្លាស់ប្តូរពីប្រមាណមួយទៅប្រមាណវិធីមួយភ្លាមៗ(ដកទៅបូក)

៥. មុខងារគណនាលេខទសភាគ៖ អាចធ្វើការគណនាលេខទសភាគបានយ៉ាងងាយស្រួល។

៦. ការរចនាមានការដំណើរការឿន៖ ធ្វាំងកម្មវិធីអាចផ្លាស់ប្តូរទៅតាមទំហំទៅតាម devicesរបស់ usersបានយ៉ាងរលូននិងរហ័ស។

៣. រចនាសម្ព័ន្ធគម្រោង

MyApp.dart: ចំណុចចូលនៃកម្មវិធី មាន class MyAppនិង HomePage ។

buttons.dart: មាន class MyButtonដែលជា stateless widgetដែលប្រើសម្រាប់បង្កើត buttonsម៉ាស៊ីនគិតលេខ

៤. ការពិពណ៌នាFiles

៤.១. MyApp.dart

```dart
import 'package:flutter/material.dart';

import 'buttons.dart';
import 'package:math_expressions/math_expressions.dart';

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      home: HomePage(),
    );
  }
}

class HomePage extends StatefulWidget {
  @override
  _HomePageState createState() => _HomePageState();
```

```dart
}

class _HomePageState extends State<HomePage> {
  var userInput = '';
  var answer = '';

  // Array of buttons
  final List<String> buttons = [
    'C',
    '+/-',
    '%',
    'DEL',
    '7',
    '8',
    '9',
    '/',
    '4',
    '5',
    '6',
    'x',
    '1',
    '2',
    '3',
    '-',
    '0',
    '.',
    '=',
    '+',
  ];

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("Calculator"),
      ),
      backgroundColor: Colors.white38,
      body: LayoutBuilder(
        builder: (BuildContext context, BoxConstraints constraints) {
          final double screenHeight = constraints.maxHeight;
          final double screenWidth = constraints.maxWidth;
          final double buttonHeight = screenHeight * 0.7 / 5;

          return Column(
            children: <Widget>[
              Container(
                height: screenHeight * 0.3, // 30% of screen height for display
                padding: EdgeInsets.all(20),
                alignment: Alignment.centerRight,
                child: Column(
```

```
            mainAxisAlignment: MainAxisAlignment.center,
            crossAxisAlignment: CrossAxisAlignment.end,
            children: <Widget>[
              Text(
                userInput,
                style: TextStyle(fontSize: 24, color: Colors.white),
              ),
              SizedBox(height: 10),
              Text(
                answer,
                style: TextStyle(
                  fontSize: 48,
                  color: Colors.white,
                  fontWeight: FontWeight.bold,
                ),
              ),
            ],
          ),
        ),
        Container(
          height: screenHeight * 0.7, // 70% of screen height for buttons
          child: GridView.builder(
            physics: NeverScrollableScrollPhysics(), // Disable scrolling
            itemCount: buttons.length,
            gridDelegate: SliverGridDelegateWithFixedCrossAxisCount(
              crossAxisCount: 4,
              childAspectRatio: screenWidth / (buttonHeight * 4),
            ),
            itemBuilder: (BuildContext context, int index) {
              // Clear Button
              if (index == 0) {
                return MyButton(
                  buttontapped: () {
                    setState(() {
                      userInput = '';
                      answer = '0';
                    });
                  },
                  buttonText: buttons[index],
                  color: Colors.blue[50]!,
                  textColor: Colors.black,
                );
              }

              // +/- button
              else if (index == 1) {
                return MyButton(
                  buttontapped: () {
                    setState(() {
                      // Implement +/- functionality here
```

```
        });
      },
      buttonText: buttons[index],
      color: Colors.blue[50]!,
      textColor: Colors.black,
    );
  }
  // % Button
  else if (index == 2) {
    return MyButton(
      buttontapped: () {
        setState(() {
          userInput += buttons[index];
        });
      },
      buttonText: buttons[index],
      color: Colors.blue[50]!,
      textColor: Colors.black,
    );
  }
  // Delete Button
  else if (index == 3) {
    return MyButton(
      buttontapped: () {
        setState(() {
          userInput =
              userInput.substring(0, userInput.length - 1);
        });
      },
      buttonText: buttons[index],
      color: Colors.blue[50]!,
      textColor: Colors.black,
    );
  }
  // Equal_to Button
  else if (index == 18) {
    return MyButton(
      buttontapped: () {
        setState(() {
          equalPressed();
        });
      },
      buttonText: buttons[index],
      color: Colors.orange[700]!,
      textColor: Colors.white,
    );
  }

  // other buttons
  else {
```

```
                    return MyButton(
                      buttontapped: () {
                        setState(() {
                          userInput += buttons[index];
                        });
                      },
                      buttonText: buttons[index],
                      color: isOperator(buttons[index])
                          ? Colors.blueAccent
                          : Colors.white,
                      textColor: isOperator(buttons[index])
                          ? Colors.white
                          : Colors.black,
                    );
                  }
                },
              ),
            ),
          ],
        );
      },
    ),
  );
}

bool isOperator(String x) {
  return ['/', 'x', '-', '+', '='].contains(x);
}

// function to calculate the input operation
void equalPressed() {
  String finaluserinput = userInput;
  finaluserinput = userInput.replaceAll('x', '*');

  Parser p = Parser();
  Expression exp = p.parse(finaluserinput);
  ContextModel cm = ContextModel();
  double eval = exp.evaluate(EvaluationType.REAL, cm);
  setState(() {
    answer = eval.toString();
  });
}
}
```

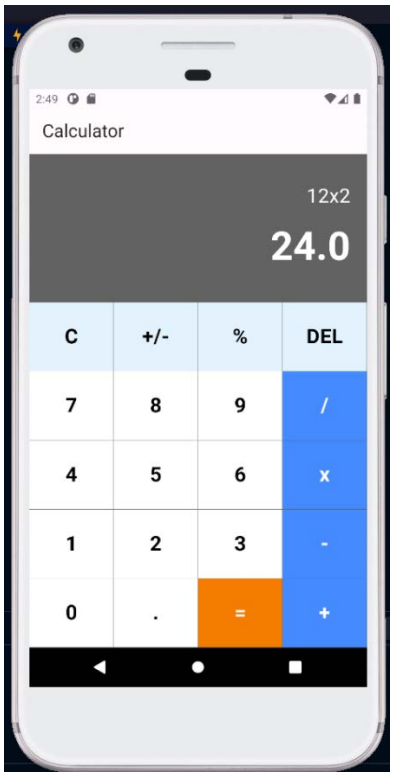
৫.৬. button.dart

```dart
import 'package:flutter/material.dart';


// creating Stateless Widget for buttons
class MyButton extends StatelessWidget {
  // declaring variables
  final Color color;
  final Color textColor;
  final String buttonText;
  final VoidCallback buttontapped;

  // Constructor
  const MyButton({
    Key? key,
    required this.color,
    required this.textColor,
    required this.buttonText,
    required this.buttontapped,
  }) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return GestureDetector(
      onTap: buttontapped,
      child: Padding(
        padding: const EdgeInsets.all(0.2),
        child: ClipRRect(
          // borderRadius: BorderRadius.circular(25),
          child: Container(
            color: color,
            child: Center(
              child: Text(
                buttonText,
                style: TextStyle(
                  color: textColor,
                  fontSize: 25,
                  fontWeight: FontWeight.bold,
                ),
              ),
            ),
          ),
        ),
      ),
    );
  }
}
```

៥. របៀបតំឡើង

1. **Install Flutter:** ត្រូវប្រាកដថា Flutter ត្រូវបានដំឡើង និងអនុវត្តតាមការណែនាំអំពីការដំឡើង Flutter ផ្លូវការ។
2. **Clone the Repository:** Clone the project repository ទៅក្នុង local machine របស់អ្នក។
3. **Open in IDE:** បើក project នៅក្នុង IDE (ឧទាហរណ៍ VS Code, Android Studio) ។
4. **Run the App:** ប្រើ IDE ដើម្បីដំណើរការកម្មវិធីនៅលើ emulator ឬឧបករណ៍ជាក់ស្តែង។

៦. ការប្រើប្រាស់



1. **Clear (C):** លុបចោលរាល់ការបញ្ចូលទាំងអស់
2. **+/−:** ផ្លាស់ប្តូរសញ្ញានៃការបញ្ចូលបច្ចុប្បន្ន
3. **%:** បន្ថែមសញ្ញាភាគរយទៅការបញ្ចូល
4. **DEL:** លុបលេខចុងក្រោយនៅក្នុងការបញ្ចូល
5. **លេខ (0-9):** បន្ថែមលេខទៅការក្នុងបញ្ចូល
6. **Operators (/ x - +):** បន្ថែម Operator ទៅនឹងធាតុបញ្ចូល
7. **Decimal (.):** បន្ថែមទសភាគទៅធាតុដែលបានបញ្ចូល
8. **Equal (=):** គណនាកន្សោមលេខដែលបញ្ចូល និងបង្ហាញលទ្ធផល

៧. ការពង្រឹងនាពេលអនាគត

1. **Advanced Functions:** បន្ថែមអនុគមន៍គណិតវិទ្យាកម្រិតខ្ពស់បន្ថែមទៀតដូចជា ឫសការ៉េ និងទស្សភាគ
2. **History:** រក្សាប្រវត្តិនៃការគណនា
3. **Themes**: បន្ថែមThemes ជាច្រើនទៅលើ Interfaceរបស់ម៉ាស៊ីនគិតលេខ