# КАЛЬКУЛЯТОР

Цифры: 1,2,3,4,5,6,7,8,9,0

Функции: «+, -, ×, ÷, %, √, ^ (на клавиатуре - хn), (, ), cos, sin, tg, π»

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| + | - | × | ÷ | % | √ | ^ | ( | ) | cos | sin | tg | π |

Значения: «000, 2×000, 3×000»

Клавиши удаления: «С, 🡨»

Клавиша «=»

Текстовое поле: “editText”

Текстовое поле: “result”

## Цифры

1. Пустая строка: просто присоединяется;
2. После функций:
   1. если функция +,-,×,÷,√, ^, (, cos, sin, tg -> просто присоединяется
   2. если функция %, ), π -> добавляется «×» (5%×2)

## Функции

### «+,-,×,÷»

1. Пустая строка: не ставятся (не активны)
2. После цифр: просто присоединяются;
3. После функций: присоединяются только после – π, )

### «cos, sin, tg»

1. Пустая строка: просто присоединяется;
2. После цифр: добавляется «×» (5,0×cos60)
3. После функций +,-,×,÷, √, ^, (: просто присоединяется;
4. После функций «), cos, sin, tg, π»: добавляется «×» (π ×cos60)

### «√, (, π»

1. Пустая строка: просто присоединяется;
2. После цифр: добавляется «×» (5,0× π)
3. После функций +, -, ×, ÷, (: просто присоединяется;
4. После функций «), cos, sin, tg, π»: добавляется «×» (π ×cos60)

### «^ , ), %»

1. Пустая строка: не ставится
2. После цифр: просто присоединяется.
3. После функций +, -, ×, ÷, √, cos, sin, tg, π: не ставится;
4. После функции «(«: ставится только «)». Добавляется только тогда, когда перед этим был символ «(»;

## Значения: «000, 2×000, 3×000»

Ставятся только после цифр!

## Клавиша «=»

При нажатии на клавишу, берется значение с “Result” вставляется в “editText”

## Клавиша «С»

editText.setText(“”)

result.setText(“0”)

## Клавиша «🡨»

1. Программа считывает положение курсора и удаляет символ перед курсором;
2. Если перед курсором функции с несколькими знаками (cos, sin), удаляет сразу всю функцию;
3. При нажатии на выражение курсор должен устанавливаться, но клавиатура телефона не должна появляться;

## Общее:

1. Все значения вводить согласно положению курсора. После ввода, курсор должен становиться после вводимого значения;
2. Необходимо добавить БД Room. Это будет отдельный экран, который будет хранить архив совершенных операций и дату выполнения данной операции. Операции будут сохраняться при нажатии на клавишу «С» или «=». При нажатии на ячейку архива можно: взять выражение, удалить выражение. Храниться 1000 выражений. Остальные удаляются.
3. Необходимо, чтобы результат выводил 8 чисел после запятой, если это число не целое;
4. Числа должны записываться по разрядам, с разделением через пробел (22 222 222);

## База данных

1. Сохранение выражения в БД идет по длинному нажатию «=»;
2. В БД длинное нажатие на список:
   1. Добавить описание;
   2. Взять выражение
   3. Удалить выражение
3. Поменять значок БД сверху в фрагменте «Калькулятор»

**private void** dialogForCurrencyPair(**final** String name) {  
 View view = getLayoutInflater().inflate(R.layout.***dialog\_for\_pairs***, **null**);  
 **final** AlertDialog alertDialog = **new** AlertDialog.Builder(**context**)  
 .setTitle(name)  
 .setView(view)  
 .create();  
  
 **final** TextView show\_graphic = view.findViewById(R.id.***show\_graphic***);  
 show\_graphic.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 GraphicFragment graphicFragment = **new** GraphicFragment();  
 **callback**.dataToActivity();  
 Bundle bundle = **new** Bundle();  
 bundle.putString(**"GRAPH"**, name);  
 graphicFragment.setArguments(bundle);  
 getFragmentManager().beginTransaction().replace(R.id.***container\_fragment***, graphicFragment)  
 .addToBackStack(**"Graph"**).commit();  
  
 alertDialog.cancel();  
 }  
 });  
  
 **final** TextView history = view.findViewById(R.id.***history***);  
 history.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 HistoryOfDatabaseFragment historyOfDatabaseFragment = **new** HistoryOfDatabaseFragment();  
 Bundle bundle = **new** Bundle();  
 bundle.putString(**"HISTORY"**, name);  
 historyOfDatabaseFragment.setArguments(bundle);  
 getFragmentManager().beginTransaction().add(R.id.***big\_fragment\_container***, historyOfDatabaseFragment)  
 .addToBackStack(**"History"**).commit();  
 alertDialog.cancel();  
 }  
 });  
  
  
 alertDialog.show();  
  
}