```
#!/bin/bash

#       *** CALCULATOR ***        ボタン#1にて  可変部シフト
# ロータリエンコーダのデータを得て  計算･表示  そして  ボタン#4押しで抜ける
#           最近  出来たばかりの  ./re_checkを採用 表示ブリンクします              2005/11/8 By  さえき

#


p=$1
if [ $# -lt 2 ]; then
 p="j_p main_menu_"
fi
        printf "\0" > /dev/ttyS1                                                        # clr
                printf "\20" > /dev/ttyS1               #for test LED on
        printf "\1\00" > /dev/ttyS1                                                     # pos(0,0)
        printf " *** CALCULATOR ***" > /dev/ttyS1

        a=$(./re_check 15 0 199 1)                                      # 初期値  15で  0～199のリミット
        b=$(./re_check 1 List: \+ \- x \/ \/ )    # 初期値    List  で  1番目
        c=$(./re_check 2 1 199 1)                                               # 初期値  2で  1～199のリミット
        sh=$(./re_check 1 List: s sh shi shif shift _KEY \#1_is  _Key shift shif shi sh)

                                val_sh=$(echo "$sh" |awk '{print $1}')
                                disp_sh=$(echo "$sh" |awk '{print $2}' |sed -e 's/_/ /g')
                                val_a=$(echo "$a" |awk '{print $1}')
                                disp_a=$(echo "$a" |awk '{print $2}' |sed -e 's/_/ /g')
                                val_b=$(echo "$b" |awk '{print $1}')
                                disp_b=$(echo "$b" |awk '{print $2}' |sed -e 's/_/ /g')
                                val_c=$(echo "$c" |awk '{print $1}')
                                disp_c=$(echo "$c" |awk '{print $2}' |sed -e 's/_/ /g')
        n=0                                                                     # 0 a , 1 b , 2 c /Re Pos  cyclic

while :;do

                printf "\3\00" > /dev/ttyS1                                                     # 3gyoume

        case $(($val_b)) in
                1 ) printf "$disp_a $disp_b $disp_c = " > /dev/ttyS1
                                printf "%d\n" $(($val_a + $val_c)) > /dev/ttyS1;;
                2 ) printf "$disp_a $disp_b $disp_c = " > /dev/ttyS1
                                printf "%d\n" $(($val_a - $val_c)) > /dev/ttyS1;;
                3 ) printf "$disp_a $disp_b $disp_c = " > /dev/ttyS1
                                printf "%5d\n" $(($val_a * $val_c)) > /dev/ttyS1;;
                4 ) if [ $disp_b = "/" ]; then      #  算数の割り算記号÷の表示にする
                                        disp_b="\xfd"
                                fi
                                printf "$disp_a $disp_b $disp_c = " > /dev/ttyS1
                                printf "%2d .." $(($val_a / $val_c)) > /dev/ttyS1
                                printf "%3d\n" $(($val_a % $val_c)) > /dev/ttyS1;;
                5 ) printf "$disp_a $disp_b $disp_c = " > /dev/ttyS1
                                printf  $(echo "scale=5; $val_a / $val_c " |bc)"\n" > /dev/ttyS1;;

        esac

                        printf "\4\00" > /dev/ttyS1                                                     # 4gyoume
                        printf "$disp_sh      " > /dev/ttyS1
                        printf "\4\22OK" > /dev/ttyS1                                           # 4gyoume \22(=18desimal)

        printf "\20" > /dev/ttyS1               #for test LED on


        while :;do
                case "$(./onebyone_S1)" in              #ボタン状態  拾う

                "N" ) n=$(($n + 1)) #button 1   RE Pos shift
                                        if [ $n -gt 3 ]; then n=0 #0 1 2 3 の繰り返し
                                        fi                                                              #シフトkey押された

        printf "\21" > /dev/ttyS1               #for test LED off

                        case $n in                      #みなやっちゃうと時間掛かる  そこで  個々必要なものだけ
                        0)      sh=$(./re_check "$sh" -R)               #  移動先のre値リセットしなければ
                                        c=$(./re_check "$c" -R)                         #  また  移動前の表示も点滅止めに
                                        disp_c=$(echo "$c" |awk '{print $2}' |sed -e 's/_/ /g')
                                ;;
                        1)      a=$(./re_check "$a" -R)
                                        sh=$(./re_check "$sh" -R 1)   #移動前の表示  特別に  元の値を再設定
                                        disp_sh=$(echo "$sh" |awk '{print $2}' |sed -e 's/_/ /g')
                                ;;
                        2)      b=$(./re_check "$b" -R)
                                        a=$(./re_check "$a" -R)                 #  また  移動前の表示も点滅止めに
#                                       a=$(./re_check "$a" -R 88)    #あわせて  特別に  値を再設定するなら
#                                       val_a=$(echo "$a" |awk '{print $1}')        #表示に加え  値も書替えよ  (注意点)
                                        disp_a=$(echo "$a" |awk '{print $2}' |sed -e 's/_/ /g')
#                       (注意点)  表示に反映させるだけならいいが    時間節約を考慮しつつも
#                       このプログラムのように計算にすぐ使うなら  すぐに変数も書き換えること

                                ;;
                        3)      c=$(./re_check "$c" -R)
                                        b=$(./re_check "$b" -R)
                                        disp_b=$(echo "$b" |awk '{print $2}' |sed -e 's/_/ /g')
                                ;;
                        esac
                        ;;
                "G" ) break 2 ;; #button 4    抜ける
                "" ) continue ;; #通信  断？    ループ持続でいいかな？  カウントしbreakすべきだな

                * )     #指定のボタン押しなし  最新RE状況を変数にまで反映させて  頭に戻す
                        case $n in              #みなやっちゃうと時間掛かる  そこで  個々必要なものだけ
                        0 ) sh=$(./re_check "$sh")
                                        disp_sh=$(echo "$sh" |awk '{print $2}' |sed -e 's/_/ /g')
                                        val_sh=$(echo "$sh" |awk '{print $1}');;
```

```
                        1 ) a=$(./re_check "$a")
                                disp_a=$(echo "$a" |awk '{print $2}' |sed -e 's/_/ /g')
                                val_a=$(echo "$a" |awk '{print $1}');;
                        2 ) b=$(./re_check "$b")
                                disp_b=$(echo "$b" |awk '{print $2}' |sed -e 's/_/ /g')
                                val_b=$(echo "$b" |awk '{print $1}');;
                        3 ) c=$(./re_check "$c")
                                disp_c=$(echo "$c" |awk '{print $2}' |sed -e 's/_/ /g')
                                val_c=$(echo "$c" |awk '{print $1}');;
                        esac
                                break ;;
                esac

        while :;do
                case "$(./onebyone_S1)" in                      #ボタン開放待ち
                "O" ) break 2 ;;        #ボタン開放になった  抜ける
                ""              ) break 2 ;;
                * ) continue ;;         #ボタン押しのままだ     ループ
                esac
                done
        done
done
        pp=$(./$p $2)
        echo "$pp"
exit
```