```
; гр. ИУ6-71 Васильев
.include "m8515def.inc" ; Используем ATmega8515
.def tmp = r16 ; Временный регистр
.def dr1 = r17 ; Регистр задержки
.def stt = r18 ; Регистр режима/выбранного диапазона
.def stp = r19 ; Регистр состояния/шага работы
.def with = r20 ; Регистр ожидания - старший десят разряд
.def witl = r21 ; Регистр ожидания - младший десят разряд
.def wrt = r22 ; Регистр записи
.def resl = r23 ; Регистр результата - старший разряд
.def resh = r24 ; Регистр результата - младший разряд
.def rep = r25 ; Кол-во повторений
.equ STEP = 0 ; Кнопка "Далее / След шаг"
.equ STAT = 1 ; Кнопка "Сменить режим / др диапазон"

.org $000
     rjmp init
.org $006
     rjmp t1_ovf

init: ; *Инициализация
     ldi tmp, low(RAMEND) ;установка
     out SPL, tmp ; указателя стека
     ldi tmp, high(RAMEND) ; на последнюю
     out SPH, tmp

     ldi tmp,0x80 ;разрешение прерывания по
     out TIMSK,tmp ; переполнению таймера T1
     clr tmp ;таймер T1
     out TCCR1B,tmp ; остановлен
     cli

     ldi tmp, 0x00
     out UCSRA, tmp
     sbi UCSRB, TXEN ; Разрешение передачи
     ;ldi tmp, 0x86 ; Формат посылки
     ;out UCSRC, tmp
     ldi tmp, 0x00
     out UBRRH, tmp
     ldi tmp, 11
     out UBRRL, tmp ; Задание скорости передачи

     ldi rep, 0b00000000
     ldi ZL, $40
     ldi ZH, $02
     ldi stt, 0b10000000 ; Начинаем в режиме 0 - интервалы от 5 до 20
     ldi stp, 0b10000000

    ldi tmp, 0b11001111 ; Настраиваем порт C на вывод на дисплей
     out DDRC, tmp
     ldi tmp, 0b00000011 ; Настраиваем порт D на вывод на UART
     out DDRD, tmp
     ldi tmp, 0b00000000 ; Настраиваем порт A на прием с кнопок
     out DDRA, tmp
     ldi tmp, 0b11111111 ; Подтягивающие резисторы
     out PORTA, tmp

     ;Первое сообщение UART 0 'hi'
```

```
ldi tmp, 'h'
out UDR, tmp
sbi UCSRA,TXC
rcall delay
ldi tmp, 'i'
out UDR, tmp
sbi UCSRA,TXC
rcall delay

;Инициализация LCD
ldi tmp, 0b00000011 ; Выводим тройку на D4-D7
out PORTC, tmp
rcall delay
sbi PORTC, 6 ; Устанавливает строб E - одна тройка принята
rcall delay
cbi PORTC, 6 ; Снимаем строб E
rcall delay
sbi PORTC, 6 ; Вторая тройка
rcall delay
cbi PORTC, 6
rcall delay
sbi PORTC, 6 ; Третья тройка
rcall delay
cbi PORTC, 6
rcall delay
;_
ldi tmp, 0b00000010
out PORTC, tmp
rcall delay
sbi PORTC, 6
rcall delay
cbi PORTC, 6
rcall delay
sbi PORTC, 6
rcall delay
cbi PORTC, 6
rcall delay
;_
ldi tmp, 0b00000001
out PORTC, tmp
rcall delay
sbi PORTC, 6
rcall delay
cbi PORTC, 6
rcall delay
;_
ldi tmp, 0b00000000
out PORTC, tmp
rcall delay
sbi PORTC, 6
rcall delay
cbi PORTC, 6
rcall delay
;_
ldi tmp, 0b00001111
out PORTC, tmp
rcall delay
sbi PORTC, 6
```

```
rcall delay
cbi PORTC, 6
rcall delay

; Сообщаем, что находимся в нулевом режиме
mov wrt, stt
ldi tmp, 0b10000011
rcall write_wrt
; Ожидаем нажатия кнопки
loop1:
      sbic PINA, STEP
      rjmp loop2
      rjmp stepn
loop2:
      sbic PINA, STAT
      rjmp loop1
      rjmp statn

statn:
      inc stt
      cpi stt, 0b10000100
      brne statn2
      ; Сбрасываем мод в 0
      ldi stt, 0b10000000
statn2:
      rcall clear_display
      mov wrt, stt
      ldi tmp, 0b10000011
      rcall write_wrt
      rjmp loop1

step_gen: ; Генерируем случайное число при помощи таймера
      cpi stt, 0b10000000
      breq step_gen1
      cpi stt, 0b10000001
      breq step_gen2
      cpi stt, 0b10000010
      breq step_gen3
      cpi stt, 0b10000011
      breq step_gen4
step_gen1: ;5 .. 20
      ldi resl, 0b10000101
      ldi resh, 0b10000000
      ldi with, 0b10000010
      ldi witl, 0b10000000
      rjmp step_gen5
step_gen2: ;20 .. 40
      ldi resh, 0b10000010
      ldi resl, 0b10000000
      ldi with, 0b10000100
      ldi witl, 0b10000000
      rjmp step_gen5
step_gen3: ;30 .. 60
      ldi resh, 0b10000011
      ldi resl, 0b10000000
      ldi with, 0b10000110
      ldi witl, 0b10000000
      rjmp step_gen5
```

```
step_gen4: ;5 .. 60
     ldi resh, 0b10000000
     ldi resl, 0b10000101
     ldi with, 0b10000110
     ldi witl, 0b10000000
     rjmp step_gen5
step_gen5:
     rcall clear_display
     ldi wrt,0xFF ; загрузка TCNT1
     out TCNT1H,wrt
     ldi wrt,0xFF
     out TCNT1L,wrt
     sei ;глобальное разрешение прерываний
     ldi wrt, 0x01
     out TCCR1B,wrt ;запуск таймера с предделителем
     rjmp loop1

stepn:
     inc stp
     cpi stp, 0b10000001
     breq step_gen
     cpi stp, 0b10000010
     breq step_shg
     cpi stp, 0b10000011
     breq step_wait
     cpi stp, 0b10000100
     breq step_res
     cpi stp, 0b10000101
     breq step_c
     rjmp loop1

step_c: ; Если шаг 5, сбрасываем в 0 и выводим мод
     ldi stp, 0b10000000
     rcall clear_display
     mov wrt, stt
     ldi tmp, 0b10000011
     rcall write_wrt
     rjmp loop1

step_shg:
     cli
     clr wrt ;останов
     out TCCR1B,wrt ; таймера T1
     mov with, resh
     mov witl, resl
     rcall clear_display
     mov wrt, resh
     ldi tmp, 0b10000011
     rcall write_wrt
     mov wrt, resl
     ldi tmp, 0b10000011
     rcall write_wrt
     mov witl, resl
     mov with, resh
     rjmp loop1

step_wait:
     ldi resl, 0b10000000
```

```
            ldi resh, 0b10000000 ; Настройка таймера T1 на режим таймера
            ldi wrt,0x1E ; загрузка TCNT1
            out TCNT1H,wrt
            ldi wrt,0xC8
            out TCNT1L,wrt
            sei ;глобальное разрешение прерываний
            ldi wrt, 0x03
            out TCCR1B,wrt ;запуск таймера с предделителем
            ldi tmp, 0b10000010
            ldi wrt, 0b10000001
            rcall write_wrt
            rjmp loop1

      step_res: ; Выводим результат
            cli
            clr wrt ;остановка
            out TCCR1B,wrt ; таймера T1
            rcall clear_display
            mov wrt, resh
            ldi tmp, 0b10000011
            rcall write_wrt
            ldi tmp, 0b10000011
            mov wrt, resl
            rcall write_wrt
            ldi tmp, 0b10000010
            ldi wrt, 0b10000000
            rcall write_wrt ; Вывод пробела " "
            ; Вычитаем из полученного ожидаемое и выводим разницу
            andi resl, 0b00001111
            andi resh, 0b00001111
            andi witl, 0b00001111
            andi with, 0b00001111
            ; Вычислим двоично в виде 10resh+resl-(10with+witl)
sub_s1:     ldi tmp, 10
            mul with, tmp
            mov with, r0
            add with, witl
sub_s2:
            mul resh, tmp
            mov resh, r0
            add resh, resl
            sub resh, with
            brmi sub_c2
            rjmp sub_s3
sub_c2: ; Переводим в прямой код, если разница < 0
            ldi dr1, 0xff
            eor resh, dr1
            inc resh
sub_s3:
            ldi dr1, 0
            ldi tmp, 0b00001010
sub_loop: ; Считаем, сколько десяток влезет в число
            cp resh, tmp
            brlo sub_out
            inc dr1
            sub resh, tmp
            rjmp sub_loop
sub_out: ; Вывод на дисплей и запись в память
```

```
            mov resl, resh
            mov resh, dr1
            andi resl, 0b00001111
            andi resh, 0b00001111
            ldi dr1, $30 ; Для кодировки цифры в ascii
            mov tmp, resl
            add tmp, dr1
            st Z+, tmp
            mov tmp, resh
            add tmp, dr1
            st Z+, tmp
            inc rep
            ori resh, 0b10000000
            ori resl, 0b10000000
            ldi tmp, 0b10000011
            mov wrt, resh
            rcall write_wrt
            ldi tmp, 0b10000011
            mov wrt, resl
            rcall write_wrt
            cpi rep, 0b00000101
            breq res_res
            rjmp loop1

res_res: ; Вывод 5 сохраненных результатов по UART
      ldi tmp, ' '
      out UDR, tmp
      sbi UCSRA,TXC
      rcall delay
      ldi tmp, 'r'
      out UDR, tmp
      sbi UCSRA,TXC
      rcall delay
      ldi tmp, 'e'
      out UDR, tmp
      sbi UCSRA,TXC
      rcall delay
      ldi tmp, 's'
      out UDR, tmp
      sbi UCSRA,TXC
      rcall delay
      ldi tmp, ':'
      out UDR, tmp
      sbi UCSRA,TXC
      rcall delay
      ldi tmp, ' '
      out UDR, tmp
      sbi UCSRA,TXC
      rcall delay
mem_loop:
      ld tmp, -Z
      out UDR, tmp
      sbi UCSRA,TXC
      rcall delay
      ld tmp, -Z
      out UDR, tmp
      sbi UCSRA,TXC
      rcall delay
```

```
      ldi tmp, ' '
      out UDR, tmp
      sbi UCSRA,TXC
      rcall delay
      dec rep
      brne mem_loop
      rjmp loop1

t1_ovf: ; На разных шагах прерывание немного разное
      cpi stp, 0b10000001
      brne t1_ovf_w ; Если пользователь засекает, то переходим
      clr tmp ; Если остались - генерируем псевдорандом
      out TCCR1B,tmp ; останавливаем таймер T1
      ldi tmp,0xFF
      out TCNT1H,tmp ; перезагрузка TCNT1
      ldi tmp,0xFF
      out TCNT1L,tmp
      ldi tmp, 0x01
      out TCCR1B,tmp ;запуск таймера с предделителем
      inc resl
      cpi resl, 0b10001010
      breq resl_ovf
      cp resh, with
      breq resh_ovf
      reti

resh_ovf: ; Переполнение старшего разряда при генерации - сброс в 00
      cpi stt, 0b10000000
      breq resh_ovf1
      cpi stt, 0b10000001
      breq resh_ovf2
      cpi stt, 0b10000010
      breq resh_ovf3
      cpi stt, 0b10000011
      breq resh_ovf4
      reti
resh_ovf1:
      ldi resl, 0b10000101
      ldi resh, 0b10000000
      reti
resh_ovf2:
      ldi resh, 0b10000010
      ldi resl, 0b10000000
      reti
resh_ovf3:
      ldi resh, 0b10000011
      ldi resl, 0b10000000
      reti
resh_ovf4:
      ldi resh, 0b10000000
      ldi resl, 0b10000101
      reti

t1_ovf_w: ; Засекаем время
      clr tmp
      out TCCR1B,tmp ; останавливаем таймер T1
      ldi tmp,0x1E
      out TCNT1H,tmp ; перезагрузка TCNT1
```

```
      ldi tmp,0xC8
      out TCNT1L,tmp
      ldi tmp, 0x03
      out TCCR1B,tmp ;запуск таймера с предделителем
      inc resl
      cpi resl, 0b10001010
      breq resl_ovf
      reti

resl_ovf: ; Перепол младш разр - увелич старш, сброс младш в 0
      ldi resl, 0b10000000
      inc resh
      reti

delay:
    dms:
        ldi  dr1, 170    ; Инициализация первого цикла
    dl1:ldi  tmp, 255    ; Инициализация второго цикла
    dl2:dec  tmp        ; Тело второго цикла
        brne dl2       ; Проверка выхождения из второго цикла (переход
при 0 в конце предыдущей операции)
        dec  dr1        ; Тело первого цикла
        brne dl1        ; Проверка выхождения из первого цикла
    dend: ret          ; Возврат

write_wrt: ; tmp - первая тетрада, wrt - вторая
           out PORTC, tmp
           rcall delay
           sbi PORTC, 6 ; Строб
           rcall delay
           cbi PORTC, 6
           rcall delay
           out PORTC, wrt
           rcall delay
           sbi PORTC, 6 ; Строб
           rcall delay
           cbi PORTC, 6
           rcall delay
           ret

clear_display:
      ldi tmp, 0b00000000 ; Очистка
      out PORTC, tmp
      rcall delay
      sbi PORTC, 6 ; Строб
      rcall delay
      cbi PORTC, 6
      rcall delay
      ldi tmp, 0b00000001 ; Установка курсора
      out PORTC, tmp
      rcall delay
      sbi PORTC, 6 ; Строб
      rcall delay
      cbi PORTC, 6
      rcall delay
      ret
```