

- Vanemlektor M.J. kurdab L.K.-le :
  - “2006 aastal räägin tudengitele mis on bitt – nemad ei saa aru !”
  - .....
  - “2017 aastal räägin tudengitele mis on bitt – nemad ikka ei saa aru !”
  - .....
  - “2024 aastal räägin tudengitele mis on bitt – mina hakkab vaikselt aru saama juba aga nemad ikka ei saa aru”

# EEM3010 SISSEJUHATUS MEHHATROONIKASSE

---

Sügis 2024

Sissejuhatus programmeerimisse 2

Martin Jaanus

NRG-308

[martin.jaanus@ttu.ee](mailto:martin.jaanus@ttu.ee) 56 91 31 93

Õppetöö : <http://isc.ttu.ee>

Õppematerjalid : <http://isc.ttu.ee/martin>

# Teemad

Programmeerimisest madala taseme keeltes (C)

- Kodeerimine (andmetüübid), arvuti matemaatika eripärad.
- Arduino programmeerimise eripära
  
- Slaididel on palju näiteid (eelkõige “vigastest” programmidest)
- **Proovige need iseseisvalt järgi !**
- Need näited aitavad mõista arvuti “hingeelu”.
- **Programmeerimisoskust ei saa õpetada – see tuleb vaid läbi kogemuste !**

Väga hea veebikeskkond, kus saab harjutada ja leida näiteid erinevate keelte kohta

<https://www.w3schools.com/>

Arduino “simulator” <https://falstad.com/circuit/avr8js/>

# C keel

- 1958 kõrgkeel ALGOL (tänapäevaste kõrgkeelte eelkäija)
- 1969 Bell Laboratories , keel B <https://en.wikipedia.org/wiki/ALGOL>
- Programmeerimiskeel, mis oleks platvormist sõltumatu aga samas kiire.
- Kõrgkeeled ei suuda garanteerida ajakriitilistes kohtades kiirust ning kompaktsust
- 1973 programmeerimiskeel C (B täiustus), standardiseeriti 1983
- C++ 1989 (objektorienteeritud programmeerimine)
- C# 2002 (.net framework)
- Väga palju sugulaskeeli (sh php, java )
- Peavad olema üsna head eelteadmised riistvarast.
- **Vahendid võivad tunduda primitiivsed, aga tegelikult on väga võimekad.**

# Arduino programmeerimise eripärad

- Arduino arenduskeskkond kasutab olemuselt C++ kompilaatorit, ehk kehtivad C/C++ keele reeglid.
- Lisaks eelnevale on olemas „mugavus“täiendused ning abifunktsioonid, mida C keele maailmas ei tunta. Lisatud definitsioonid liidetakse programmi kompileerimisel. Kui on plaanis tulevikus hakata reaalses C/C++ keeles programmeerima, tasuks nendega äraharjumist vältida !
- Reeglina värvitud need keskkonnas oranžiks (aga ei pruugi)

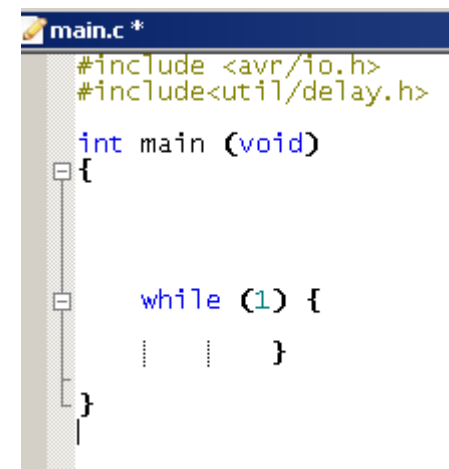
# Arduino programm (sketš)

Arduino koodis peavad olema ilmtingimata funktsioonid **setup** () , ehk kood täidetakse ühe korra ning **loop** () , kus koodi täidetakse tsüklis . Vastasel juhul tuleb kompileerimisel veateade !

Arduino „keele“ kirjeldus : <https://www.arduino.cc/reference/en/>



```
sketch_nov05a | Arduino 1.6.7
File Edit Sketch Tools Help
sketch_nov05a
void setup() {
  // put your setup code here, to run once:
}
void loop() {
  // put your main code here, to run repeatedly:
}
```

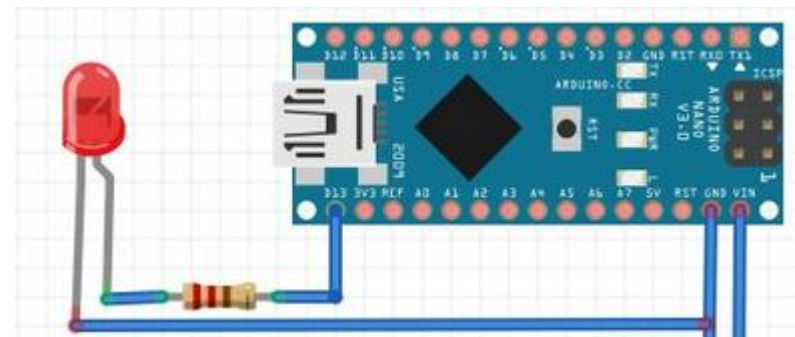


```
main.c*
#include <avr/io.h>
#include <util/delay.h>
int main (void)
{
  while (1) {
    | | }
  }
}
```

C keelne „tühi“ programm

# Arduino koodinäide

- Vilgutame valgusdiodi



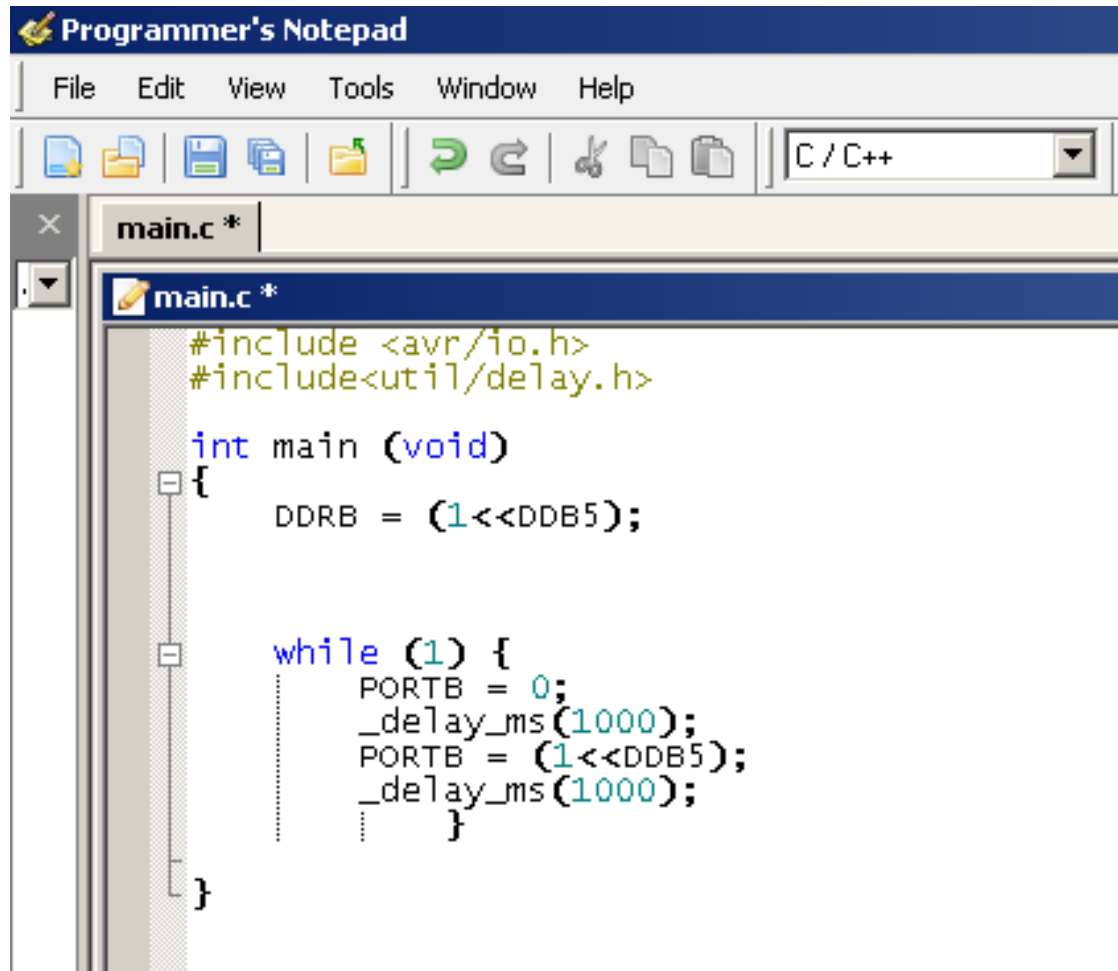
```
Blink | Arduino 1.6.7
File Edit Sketch Tools Help

Blink §

void setup() {
  // initialize digital pin 13 as an output.
  pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(13, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000);           // wait for a second
  digitalWrite(13, LOW); // turn the LED off by making the voltage LOW
  delay(1000);           // wait for a second
}
```

# C koodinäide, sama mikrokontroller



The image shows a screenshot of a text editor window titled "Programmer's Notepad". The window has a menu bar with "File", "Edit", "View", "Tools", "Window", and "Help". Below the menu bar is a toolbar with various icons for file operations and a dropdown menu set to "C/C++". The main editing area shows a file named "main.c" with the following C code:

```
#include <avr/io.h>
#include <util/delay.h>

int main (void)
{
    DDRB = (1<<DDB5);

    while (1) {
        PORTB = 0;
        _delay_ms(1000);
        PORTB = (1<<DDB5);
        _delay_ms(1000);
    }
}
```



# Assembleri koodinäide, sama mikrokontroller

- Näide tehtud õppeaine „Mikroprotsessorsüsteemid“ jaoks
- See on vaid funktsiooni väljakutsumine ja kohustuslik programmi „kest“



```
assembler_blink | Arduino 1.6.7
File Edit Sketch Tools Help
✓ → 📄 ⬆ ⬇
assembler_blink$ blink.S
extern "C" {
    // function prototypes
    void start();
;
}

void setup() {
    start();
}

void loop() {
|
}
```

# Assembleri koodinäide, sama mikrokontroller

```
assembler_blink - blink.S | Arduino 1.6.7
File Edit Sketch Tools Help
assembler_blink$ blink.S$
; Blink LED on PB5(Arduino Uno pin 13)

#define __SFR_OFFSET 0
#include "avr/io.h"
.global start
start:
; the code to execute
sbi DDRB,5 ; Set PB5 as output
LOOP1:
    sbi PORTB, 5 ; switch off the LED
    rcall DELAY_05 ; wait for half a second
    cbi PORTB, 5 ; switch it on
    rcall DELAY_05 ; wait for half a second
    rjmp LOOP1 ; jump to loop

DELAY_05: ; the subroutine:
    ldi r16, 31 ; load r16 with 31
OUTER_LOOP: ; outer loop label
    ldi r24,0 ; load registers r24:r25 with 1021, our new
                ; init value
    ldi r25, 0 ; the loop label
DELAY_LOOP: ; "add immediate to word": r24:r25 are
                ; incremented
    adiw r24, 1 ; if no overflow ("branch if not equal"), go
                ; back to "delay_loop"
    brne DELAY_LOOP
    dec r16 ; decrement r16
    brne OUTER_LOOP ; and loop if outer loop not finished
    ret ; return from subroutine
```



# Arduino (C/C++) keele süntaks

- Tehakse vahet väiksetel ja suurtähtedel ! Identifikaatorid ei tohi kokku langeda reserveeritud sõnadega (vt. Arduino reference, neid on rohkem kui C keeles, keskkond värvib need ära) [www.arduino.cc](http://www.arduino.cc)
- Näiteks `x=2;` ja `X=2;` on erinevad laused.
- Lihtlause lõpeb semikooloniga ;
- näit `if (a>b) x=2;`
- Liitlause pannakse looksulgude vahele {}, koosneb lihtlausestest.
- Näit `if (a>b) {x=2; y=3;}`
- Kommentaarid, kui ühel real, eraldatakse // , toimib kuni reavahetuseni. Pikemad kommentaarid algavad /\* ja lõppevad \*/ . Kõik, mis nende vahel, kompilaator ignoreerib.

# Arduino (C/C++) keele süntaks

- Muutuja (variable) on andmete (arvud, tekst) jaoks eraldatud mälu .
- Muutujad peavad olema defineeritud
- Saab defineerida alguses, siis eraldatakse kohe mälu.
- Saab deklareerida ka lauses, funktsioonis, siis eraldatakse mälu, aga võetakse mälu pärast selle täitmist.
  
- Näit `unsigned short MinuMuutuja;`
- Näit `float MinuMuutuja, SinuMuutuja, x, y;`
  
- Muutuja defineerimisel eraldatakse sellele mälu, milles võib olla suvaline sisu !  
Et algväärtustada muutujat, saab teha näiteks nii :
- Näit `unsigned short MinuMuutuja=0;`

# Kuidas arvutisse info on kodeeritud ?

- Peaaegu kõik tänapäevased arvutid töötavad kahendsüsteemis
- Kõige väiksem infoühik on bitt. (0 - 1, tõde-vale jms)
- Andmetüübiks bool (Boole algebra järgi) C-s puudub, alates C++
- Kõik väärtused mis on 0 on „false“, muul juhul on „true“
- `bool x=false; bool y=true;`
- 
- Kuigi bool on vaid 1 bitt , võtab see mälu ära 1 baidi (8 bitti )

Kõrgkeeles programmeerija **ei pea teadma**, kuidas arvuti töötab, insenerid, kes on seotud sardsüsteemide väljatöötamisega **võiks seda teada** ! Neil endil on lihtsam.

# Arvusüsteemidest

- On olemas positsioonilised ja mittepositsioonilised arvutussüsteemid.
- Mittepositsiooniline → Rooma süsteem.
- Positsioonilises süsteemis on tähtis numbri asukoht arvus.
- Suvaline arv  $X$  positsioonilises süsteemis alusega  $q$  üldjuhul võib esitada
- $X_q = X_{n-1} * q^{n-1} + X_{n-2} * q^{n-2} + \dots + X_0 * q^0 + X_{-1} * q^{-1} + X_{-m} * q^{-m}$
- Kus  $X_i$  on järgutegur ( $X_i = 0 \dots q - 1$ )
- $q^i$  on kaalutegur ja  $q$  on süsteemi alus
- Igapäevaelus kasutame valdavalt süsteemi  $q \rightarrow 10$
- Kuid...osade mõõtühikute puhul (näiteks aeg, ka 12, 24 ja 60 süsteemi)
- Digitaaltehnikas  $q \rightarrow 2$  (binary 0,1) , 16 (hexadecimal 0..F) , 8 (octal 0..7))
- Teoreetiliselt on võimalik suvalise arvu (sh kompleksarvu või irratsionaalarvu nt  $e$  või  $\pi$ ) baasil .

# Arvusüsteemidest

- Näide 10 süsteemi naturaalarv  $242 = 2 * 10^2 + 4 * 10^1 + 2 * 10^0$
- Teisendame 16 süsteemi (numbrid 0...F 10 - A, 11 - B, 12 - C ,13 - D,14 - E, 15 - F)
- Selleks jagame (leiame täisosa ja jäägi ) arvusüsteemi alusega (16)
- $242:16= 15$  , jääk  $2 \quad 16^0$
- $15:16=0$ , jääk  $15 \quad 16^1$
- Jagame, kuni jagatis on 0.
- Tulemus moodustub jääkidest ehk  $15 * 16^1 + 2 * 16^0 = F2$  (hex)
- 15 asemel kirjutame F



# Arvusüsteemidest

- Näide 10 süsteemi naturaalarv  $242 = 2 * 10^2 + 4 * 10^1 + 2 * 10^0$
- Teisendame 2 süsteemi (numbrid 0-1 )
- Selleks jagame (leiame täisosa ja jäägi ) arvusüsteemi alusega (2)
- $242:2= 121$  , jääk  $0$   $2^0$
- $121:2= 60$  , jääk  $1$   $2^1$
- $60:2= 30$  , jääk  $0$   $2^2$
- $30:2= 15$  , jääk  $0$   $2^3$
- $15:2= 7$  , jääk  $1$   $2^4$
- $7:2= 3$  , jääk  $1$   $2^5$
- $3:2= 1$  , jääk  $1$   $2^6$
- $1:2= 0$  , jääk  $1$   $2^7$
- Jagame, kuni jagatis on 0. Siit näha, et vaja 8 bitti .Tulemus moodustub jääkidest ehk
- $1 * 2^7 + 1 * 2^6 + 1 * 2^5 + 1 * 2^4 + 0 * 2^3 + 0 * 2^2 + 1 * 2^1 + 0 * 2^0 = 11110010$  (bin)

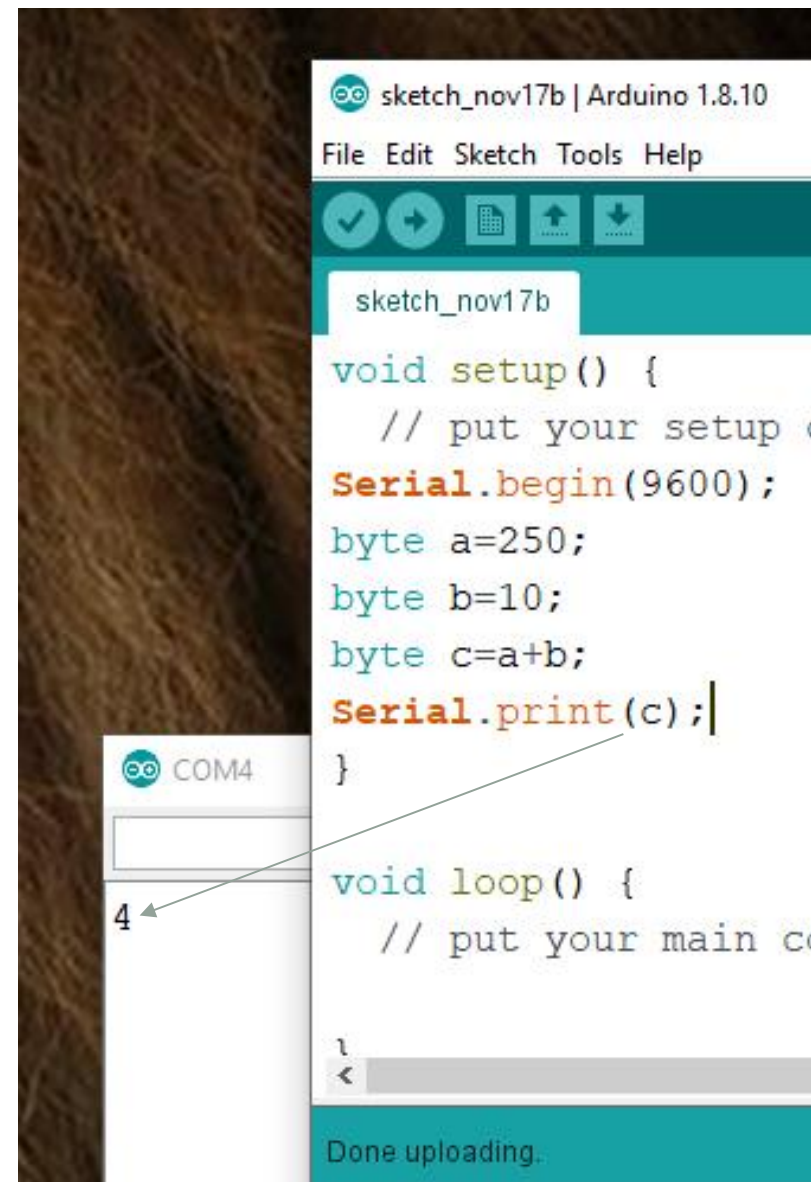
# Kahendsüsteem ja kuueteiskümnensüsteem

- On omavahel lihtsalt seotud ( $16 = 2^4$ ) ja seepärast arvutites kasutusel. Tuleb vaid meelde jätta see tabel →
- Näide F2 (hex) kahendüsteemi
- F asemele 1111 ja 2 asemele 0010
- $F2(\text{hex}) = 11110010(\text{bin})$
- $2F\ 6A\ 33(\text{hex}) \rightarrow 00101111\ 01101010\ 00110011$
- Saab ka vastupidi lihtsalt teisendada
- $10110011 =$  (eraldame 4 kaupa bitid)  $1011\ 0011 = B3(\text{hex})$
- Kümnenndsüsteemi sedasi teisendada lihtsalt ei saa.

BIN	HEX	DEC
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	A	10
1011	B	11
1100	C	12
1101	D	13
1110	E	14
1111	F	15

# Positiivsed täisarvud

- Vajalik bittide arv on vähemalt  $\text{int}(\log_2(n)) + 1$ , kus  $n$  on kujutatav arv, näiteks arvu 14 jaoks on vaja nelja bitti:
- $14(\text{dec}) = 2^3 2^2 2^1 2^0 = 1 1 1 0$
- Bitijada pikkused on arvutis standardiseeritud:
- 8 bitti (bait, byte), 16 bitti (sõna, word), 32 bitti (topeltsõna double word).
- Võimalike väärtuste arv  $2^n$ , aga kuna loendatakse nullist, on maksimum  $2^n - 1$
- Kui on kasutusel 8 bitti, ehk 1 bait andmeid, saab esitada  $2^8 = 256$  väärtust, ehk väärtused 0-255
- Arv  $14(\text{dec}) = 00001110\text{b}$
  
- Andmetüüp – **byte (Arduino)**, **unsigned char (C)** - 0-255
- Andmetüüp – **unsigned int** 0-65535, kaks baiti
- Andmetüüp – **unsigned long int** 0 – 4294967295, neli baiti
- On võimalikud ka pikemad tüübid.



The image shows a screenshot of the Arduino IDE interface. At the top, the title bar reads "sketch\_nov17b | Arduino 1.8.10". Below it is a menu bar with "File", "Edit", "Sketch", "Tools", and "Help". A toolbar contains icons for checking, running, and saving. The main editor area shows the following code:

```
sketch_nov17b
void setup() {
    // put your setup code here, to initialize variables
    Serial.begin(9600);
    byte a=250;
    byte b=10;
    byte c=a+b;
    Serial.print(c);
}

void loop() {
    // put your main code here, to run repeatedly
}
```

Below the editor is a serial monitor window titled "COM4". It shows a single character "4" in the input field, with a cursor at the end. The status bar at the bottom of the IDE says "Done uploading."

Mitte midagi pole siin valesti !

# Negatiivsed (märgiga) täisarvud

- Kuigi eraldi märgibitt ning arvubitid tundub loogiline, on see halb (arvutusloogika läheks keeruliseks )
- Esitatakse täiendkoodis (**Two's complement** ).
- Täiendkood: Sama positiivne arv , vahetame nullid ühtedeks ning ühed nullideks ja liidame juurde 1 (sisuliselt teeme nullkoha nihutamise)
- Arv +14(dec)= 00001110b
- Arv -14(dec)= 11110001b+00000001b=11110010b

- Kontroll 00001110b
- +1111 0010b
- -----
- (1)00000000b

 Ülekanne (Carry), kui ei võeta spetsiaalselt midagi ette, läheb see kaduma !

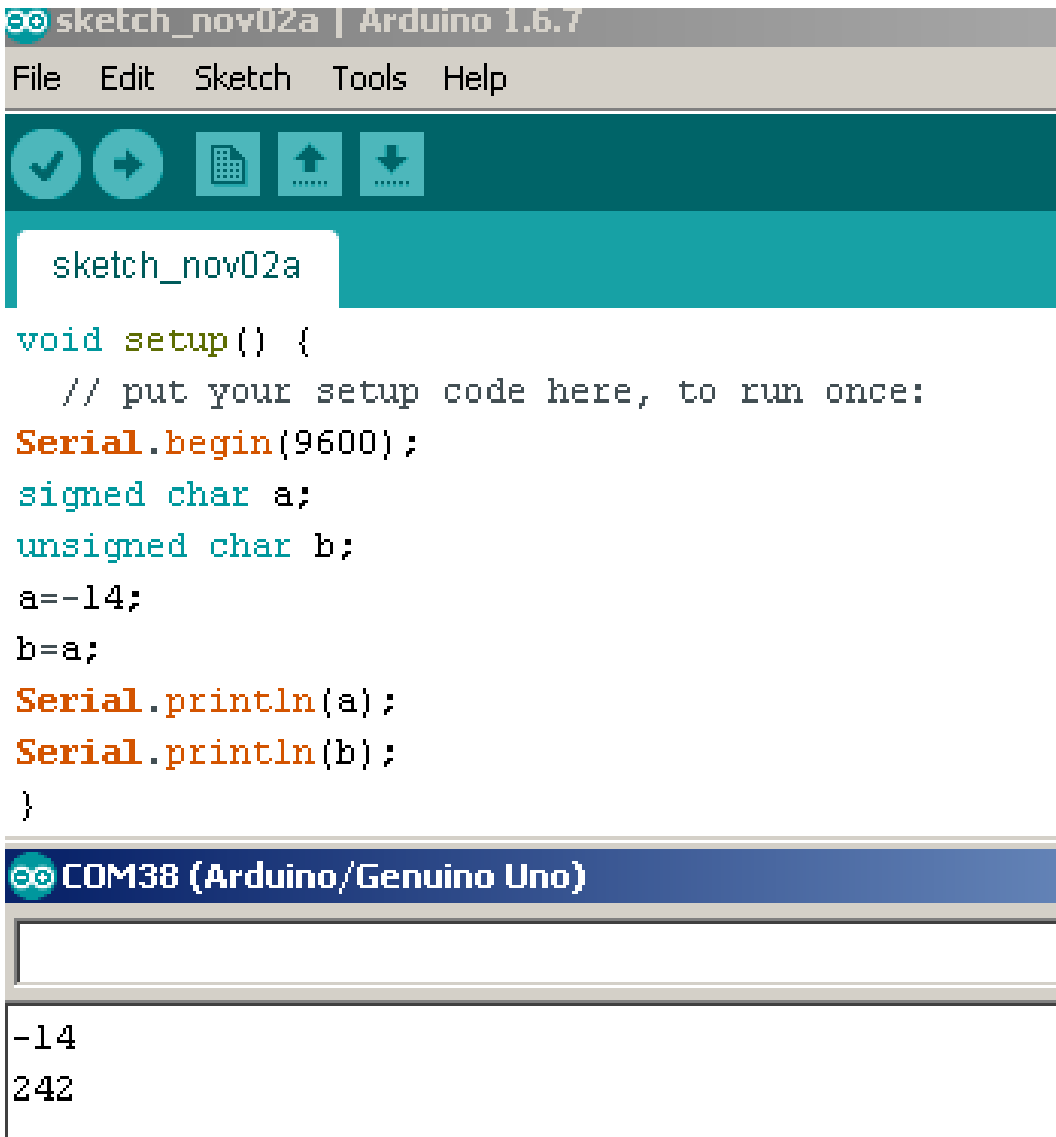
# Negatiivsed (märgiga) täisarvud

**NB !** Arv -14 võib olla aga ka positiivne arv 242 !!!  
Arvuti ei tee sellel vahet !, vahet peab tegema programmeerija !

Kui on meil teada arv positiivsel kujul ja on teada, et see esitatud täiendkoodis, siis saame negatiivse arvu lahutades positiivsest arvust  $2^n$ , kus  $n$  on bittide arv.

*Baidi puhul näiteks  $2^8 = 256$   $242 - 256 = -14$  (dec)*

- Andmetüüp **signed char** **-128...127** (C, Arduino)
- Andmetüüp **int** (Arduino) **signed short int** (C)
- **-37268...37267**
- Andmetüüp **long** (Arduino) **signed long int** (C)
- **-2147483648.... 2147483647**



The screenshot shows the Arduino IDE interface. The top menu bar includes File, Edit, Sketch, Tools, and Help. Below the menu is a toolbar with icons for saving, running, and uploading. The main editor area shows a sketch named 'sketch\_nov02a' with the following code:

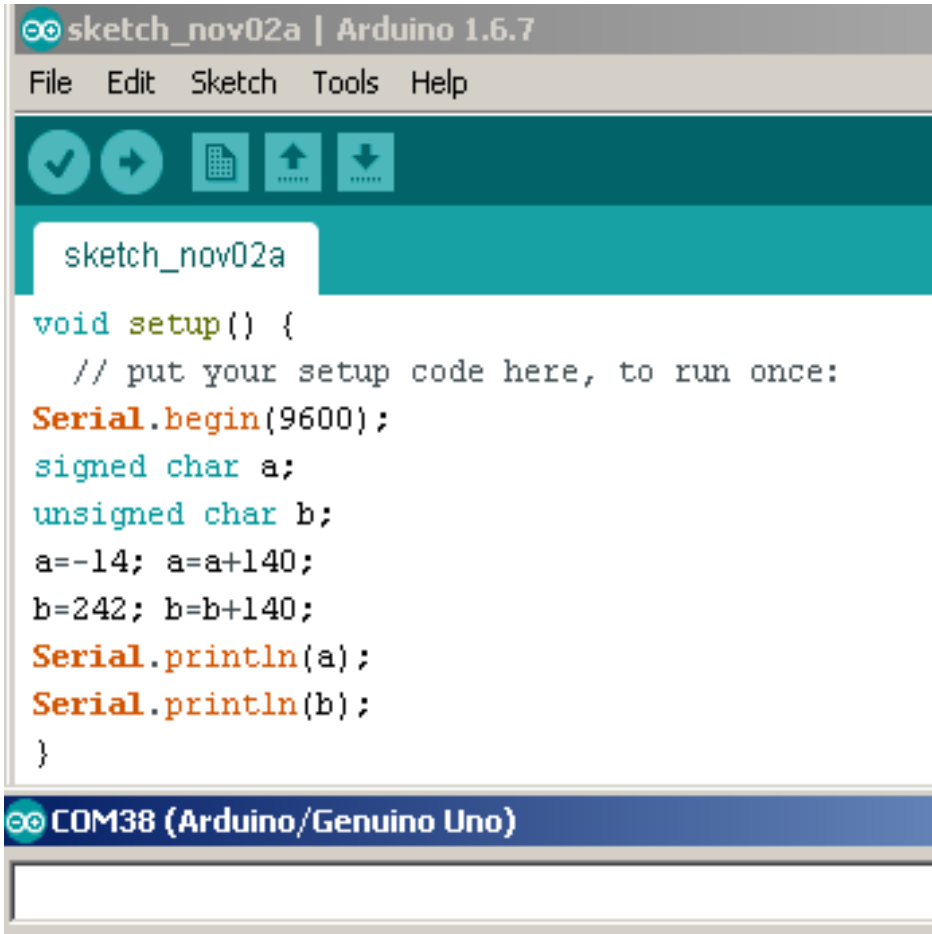
```
void setup() {  
  // put your setup code here, to run once:  
  Serial.begin(9600);  
  signed char a;  
  unsigned char b;  
  a=-14;  
  b=a;  
  Serial.println(a);  
  Serial.println(b);  
}
```

At the bottom, the serial monitor window is open, showing the output of the sketch:

```
-14  
242
```

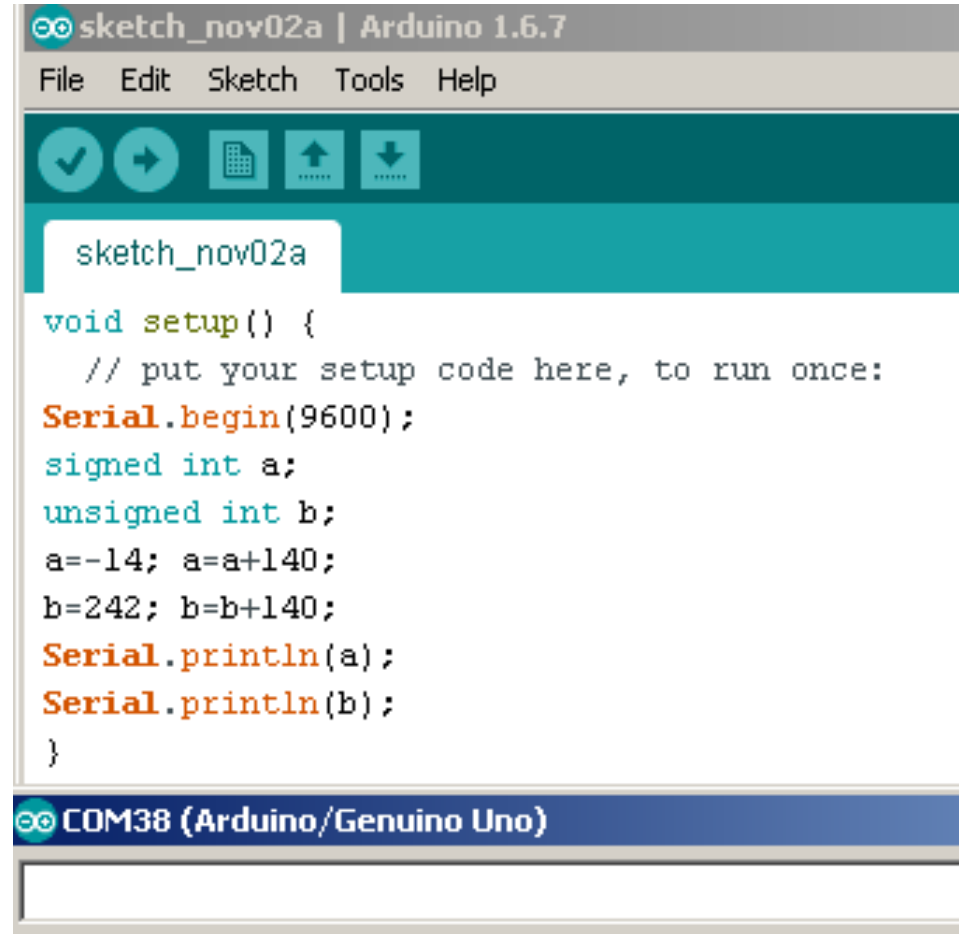
# Kui ületame piire

- ...ei anna madalataseme keeled veateadet. Arvuti teeb seda, mida tal kästakse teha !



```
sketch_nov02a | Arduino 1.6.7
File Edit Sketch Tools Help
sketch_nov02a
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  signed char a;
  unsigned char b;
  a=-14; a=a+140;
  b=242; b=b+140;
  Serial.println(a);
  Serial.println(b);
}
COM38 (Arduino/Genuino Uno)
126
```

126



```
sketch_nov02a | Arduino 1.6.7
File Edit Sketch Tools Help
sketch_nov02a
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  signed int a;
  unsigned int b;
  a=-14; a=a+140;
  b=242; b=b+140;
  Serial.println(a);
  Serial.println(b);
}
COM38 (Arduino/Genuino Uno)
382
```

126

382

# Negatiivsed (märgiga) täisarvud

- Kuigi teatud „probleemidest“ päästaks pikemad admetüübid, on neil ka suur puudus – nende kasutamine vajab rohkem mälu ning **võtab rohkem protsessori aega !**
- See võib olla väga oluline kui mälumaht ning kiirus on piiratud või ajakriitiline (mikrokontrollerid)
- Kui on selge, et lühema andmetüübiga saab hakkama, siis tuleks kasutada seda.



Attiny13A andmelehest

- High Endurance Non-volatile Memory segments
  - 1K Bytes of In-System Self-programmable Flash pro
  - 64 Bytes EEPROM
  - 64 Bytes Internal SRAM
  - Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
  - Data retention: 20 Years at 85°C/100 Years at 25°C (s
  - Programming Lock for Self-Programming Flash & E
- Peripheral Features

# Ujukomaarvud

- Ujukomaarv on paigutatav arvuti mällu kolme komponendi, märgi, arvu tüve ehk mantissi ja astendaja ehk eksponendiga.  $märk * tüvi * k^{astendaja}$
- Erinevad meetodid, levinuim IEEE 754 formaat,
- $(-1)^s * (1.M) * 2^{E-Bias}$
- Normaliseeritud arvu esimene bitt on alati 1, seda ei salvestata
- Täpsus  $1.18 \times 10^{-38}$  kuni  $3.4 \times 10^{38}$ , säilitab 6 tüvenumbrit
- , 32 bitti pikk
- Topelttäpsus  $2.23 \times 10^{-308}$  to  $1.8 \times 10^{308}$  12 tüvenumbrit, 64 bitti pikk



# Ujukomaarvud

- Single float



0 000 0000 0 000 0000 0000 0000 0000 0000 = 0.0  
0 011 1111 1 000 0000 0000 0000 0000 0000 = 1.0  
1 011 1111 1 011 0000 0000 0000 0000 0000 = -1.375  
1 111 1111 1 111 1111 1111 1111 1111 1111 = NaN

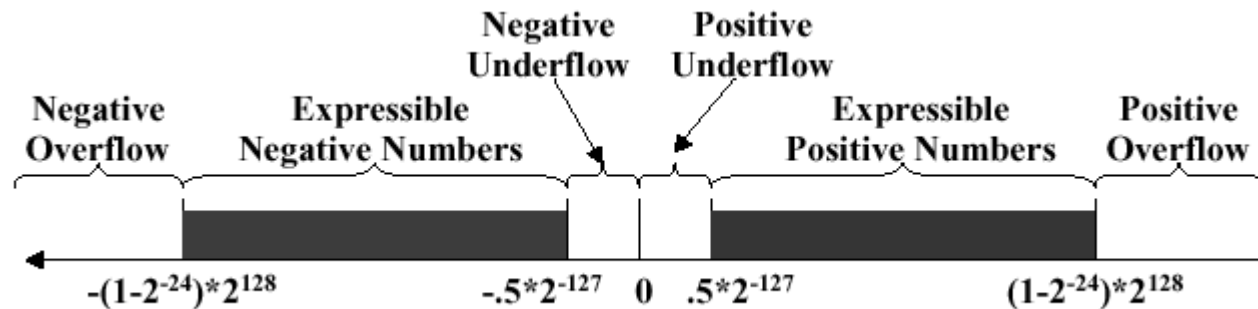
- Double float



0 000 0000 0000 0000 0000 ... 0000 0000 0000 = 0.0  
0 011 1111 1111 0000 0000 ... 0000 0000 0000 = 1.0  
1 011 1111 1110 0110 0000 ... 0000 0000 0000 = -0.6875  
1 111 1111 1111 1111 1111 ... 1111 1111 1111 = NaN

# Ujukomaarvud

- Mikrokontrolleri puhul kasutage vaid siis kui muudmoodi ei saa (kui ei ole protsessoril ujukoma arvutuse tuge, tehakse tehted tarkvaraliselt, aga see võtab aega ! )
- Probleemid : ületäitumine, alatäitumine
- Probleme tekitavad ka erinevas suurusjärgus olevate arvude liitmine-lahutamine



# Ujukomaarvud

```
COM38 (Arduino/Genuino Uno) sketch_nov02a $  
float a;  
void setup() {  
  // put your setup code here, to initialize variables  
  Serial.begin(9600);  
  float a;  
  a=10;  
  Serial.println(a);  
  a=a*a; a=a+1;  
  Serial.println(a);  
  a=a*a;  
  Serial.println(a);  
  a=a+1;  
  Serial.println(a);  
  a=a*a;  
  Serial.println(a);  
  a=a+1;  
  Serial.println(a);  
  a=a*a;  
  Serial.println(a);  
  a=/1000;  
  Serial.println(a);  
}
```

```
10.00  
101.00  
10201.00  
10202.00  
104080800.00  
104080800.00  
ovf  
ovf
```

- Kui näiteks teha FOR tsükkel , inkrementiga 1, siis ei pruugi see lõpetada !

# Matemaatika

- Mikrokontrollerite eripära – kõik tehted ei ole riistvaraliselt toetatud.
- Korrutamise-jagamise võidakse samuti teha tarkvaraliselt (näiteks TinyAVR ei oska korrutada-jagada) – aeglane (asendatakse liitmis-lahutamistsüklitega)
- Ületäitumiste eest ei hoiatata -

**Programm töötab vigase tulemusega edasi.**

(tegelikult riistvaralise tehte puhul protsessoris staatuse lippude väärtust muudetakse). C keel neid ignoreerib. Saab tekitada katkestuse.

Jagamine nulliga !

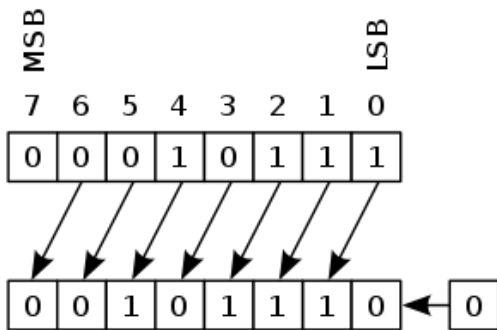
```
sketch_nov02a
float a;
void setup() {
  // put your setup code
  Serial.begin(9600);
  byte a;
  a=25;
  Serial.println(a);
  a=a*5;
  Serial.println(a);
  a=a*5;
  Serial.println(a);
}

sketch_nov02a
float a;
void setup() {
  // put your setup code
  Serial.begin(9600);
  byte a;
  a=25;
  Serial.println(a);
  a=a/5;
  Serial.println(a);
  a=a/0;
  Serial.println(a);
}
```

Tulemus ei mahu ära, lõigatakse esimesed bitid ära

# Nihe

- Korrutamine-jagamine on ebamugavad tehted mikrokontrolleritele, välja arvatud kordajaga 2
- Kümne süsteemis nihe vastab 10ga korrutamisele-jagamisele.
- Nihe vasakule suurendab väärtust kaks korda, paremale vähendab kaks korda.
- Aritmeetiline nihe – märgibitti ei nihutata
- Loogiline nihe , asemele kirjutatakse nullid
- Ringnihe – asemele kirjutatakse välja langev(ad) bit(id)



Kui võimalik , asendatakse korrutamine-jagamine nihutamiselega. Õige algoritmi ja kordajate valik

```
sketch_nov02a
void setup() {
  // put your setup
  Serial.begin(9600);
  byte a;
  a=4;
  Serial.println(a);
  a=a>>1;
  Serial.println(a);
  a=a<<3;
  Serial.println(a);
}
```

COM38 (Ar)

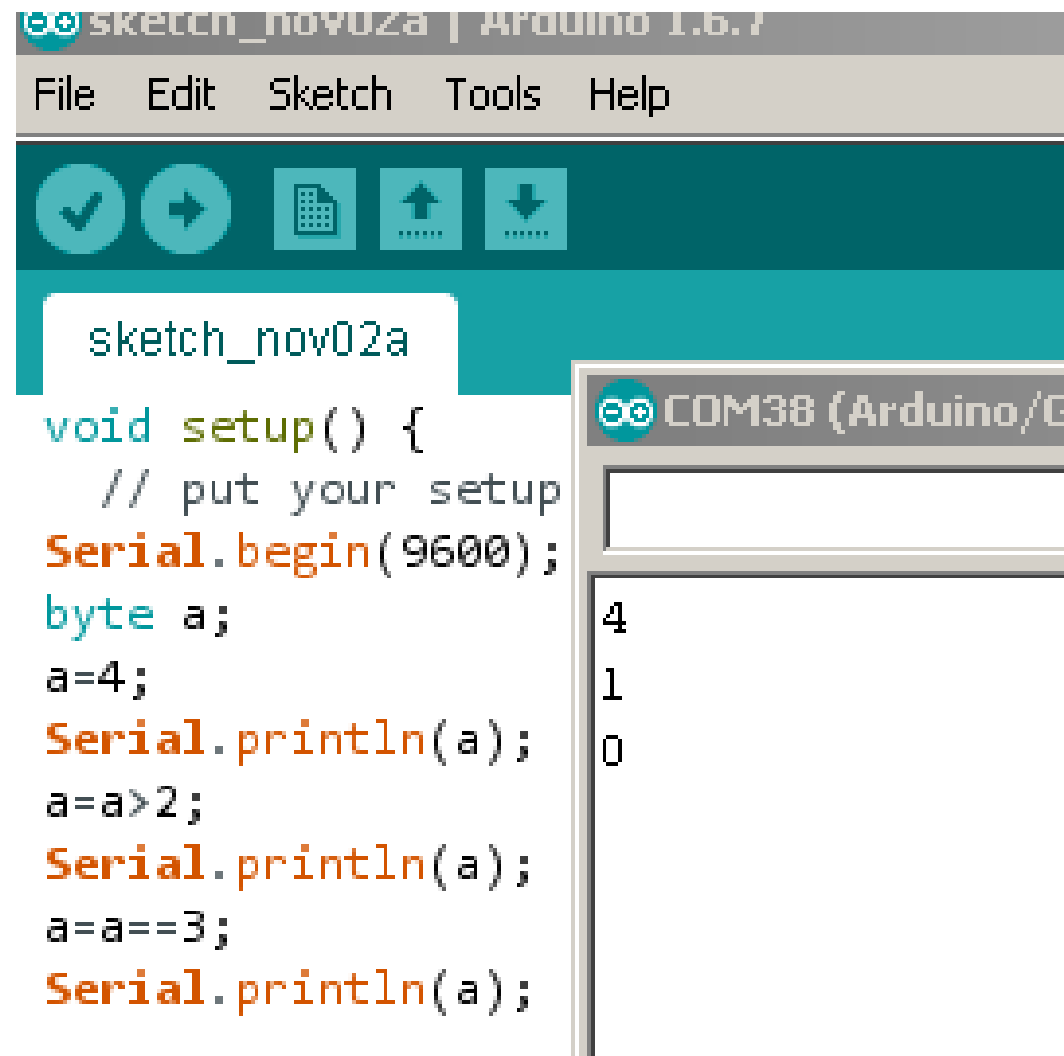
4  
2  
16

# Võrdlustehted

- == (võrdne) != (ei ole võrdne) < (väiksem kui) > (suurem kui)
- <= (väiksem võrdne kui)
- >= (suurem võrdne kui)
- Tehte tulemus on signed int, väärtusega 1, kui väide on tõene või 0 kui väide on vale

**Mitte segi ajada = ja ==  
= on omistamine !**

Osades keeltes on = nii võrdne kui ka omistamine



```
sketch_nov02a
void setup() {
  // put your setup
  Serial.begin(9600);
  byte a;
  a=4;
  Serial.println(a);
  a=a>2;
  Serial.println(a);
  a=a==3;
  Serial.println(a);
}
```

COM38 (Arduino/C

```
4
1
0
```

# Omistamine ,inkrement ja dekrement

- Omistamine (=) on sama harilik tehe nagu teised. Avaldis võib sisaldada mitu omistamistehet.
- Omistamine on võimalik vaid siis kui omistamismärgist vasakule jääb kindla mäluväljaga muutuja (ei tohi näiteks  $2=a;$ )
- Saab teha tänu sellele üsna omapäraseid lahendusi:
- $c=a+b;$   $d=e+c;$  aga sama on ka  $d=e+(c=a+b);$
- Kui muutujad on eri tüüpi ,kaasnevad tehetega **tüübiteisendused** , mis võivad omakorda tekitada segadust.
- Kui muutujate väärtuseid on vaja muuta vaid ühe võrra, saab avadlise kirjutada lihtsamalt
- $a=a+1;$  asemel  $a++;$
- $a=a-1;$  asemel  $a--;$
- Need käsud on toetatud ka riistvaraliselt.

# Tehete järjekord

- Igas programmeerimiskeeles on oluline sellele mõistetaav tehete järjekord
- C/C++ oma, kehtib ka Arduino keskkonnas
- <https://www.viva64.com/en/t/0064/>

Precedence	Operator	Description
<b>1</b>	::	Scope resolution
<b>2</b>	++ --	Suffix/postfix increment and decrement
	type() type{}	Function-style type cast
	()	Function call
	[]	Array subscripting
	. ->	Element selection by reference Element selection through pointer
<b>3</b>	++ --	Prefix increment and decrement
	+ -	Unary plus and minus
	! ~	Logical NOT and bitwise NOT
	(type)	C-style type cast
	*	Indirection (dereference)
	&	Address-of
	sizeof	Size-of
	new, new[] delete, delete[]	Dynamic memory allocation Dynamic memory deallocation
<b>4</b>	.* ->*	Pointer to member
<b>5</b>	* / %	Multiplication, division, and remainder
<b>6</b>	+ -	Addition and subtraction
<b>7</b>	<< >>	Bitwise left shift and right shift
<b>8</b>	< <=	For relational operators < and ≤ respectively
	> >=	For relational operators > and ≥ respectively
<b>9</b>	== !=	For relational = and ≠ respectively
<b>10</b>	&	Bitwise AND
<b>11</b>	^	Bitwise XOR (exclusive or)
<b>12</b>		Bitwise OR (inclusive or)
<b>13</b>	&&	Logical AND
<b>14</b>		Logical OR
<b>15</b>	?:	Ternary conditional
	=	Direct assignment (provided by default for C++ classes)
	+= -=	Assignment by sum and difference
	*= /= %=	Assignment by product, quotient, and remainder
	<<= >>=	Assignment by bitwise left shift and right shift
	&= ^=  =	Assignment by bitwise AND, XOR, and OR
<b>16</b>	throw	Throw operator (for exceptions)
<b>17</b>	,	Comma



# Massiiv (andmete kogum)

- Mugav on koondada sarnaseid elemente
- muutuja tüüp , nimi[elementide arv]
- Võib olla ka mitme dimensiooniga
- Näit byte a[10][20];

```
sketch_nov02a | Arduino 1.6.7
File Edit Sketch Tools Help

sketch_nov02a
void setup() {
  // put your setup
  Serial.begin(9600);
  byte a[10];
  byte b;
  a[1]=65;
  Serial.println(a[1])
  b=a[1];
  Serial.println(b);
  Serial.println(a[2])
}
```

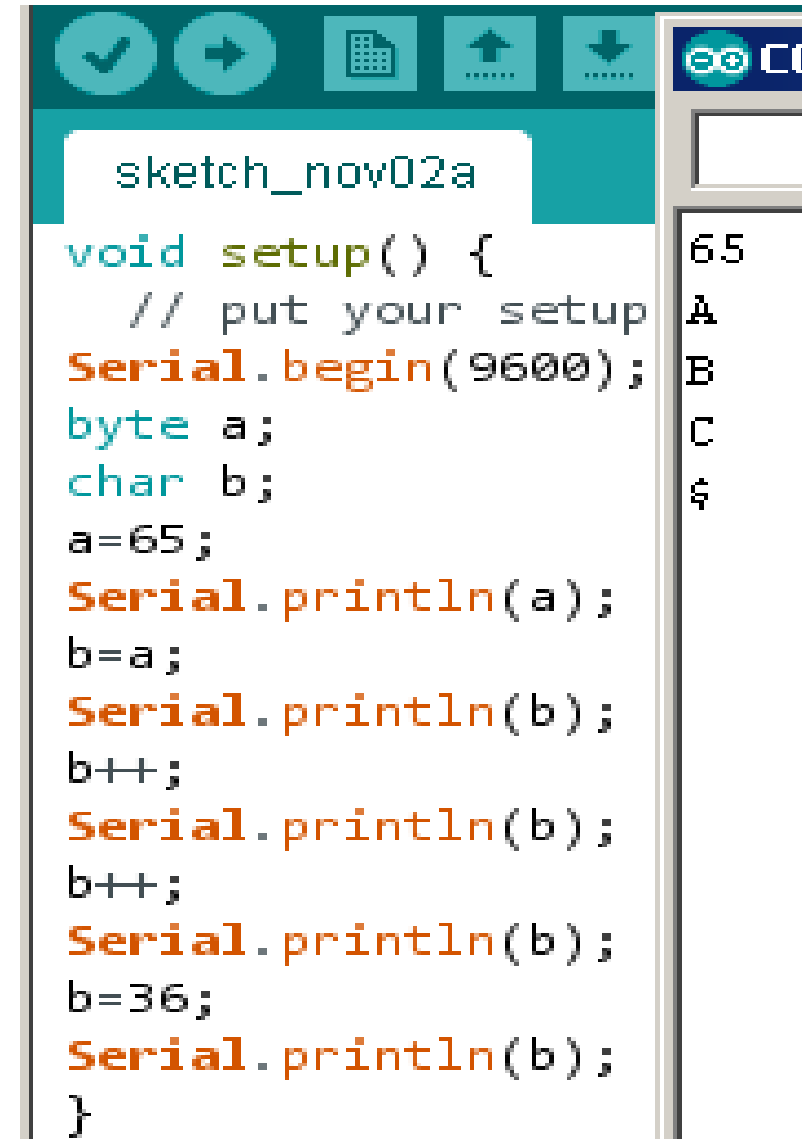
```
COM38 (Arduino/Genuino U
65
65
0
```

```
sketch_nov02a
void setup() {
  // put your setup code
  Serial.begin(9600);
  byte a[10][20];
  byte b;
  byte c=1;
  a[c][2]=65;
  Serial.println(a[c][2]);
  b=a[1][2];
  Serial.println(b);
  Serial.println(a[2][1]);
}
```

```
COM38 (Arduino
65
65
0
```

# Tekst (sümbolid)

- Numbriline väärtus vastab tähemärgile – tekstikodeering.
- Levinuim ASCII (American Standard Code for Information Interchange) kodeering 1963. aastast, põhineb inglise tähestikul .
- 7 bitiline kodeering , esimesed 28 märki on tehnilised.
- Olemuselt on tähemärk ühebaidine arv, millega saab teha kõiki tehteid.
- Arduino kompilaator teeb vahet, kas tüüp on byte või char ja vaid sedagi vaid „print“ funktsioonis , C keeles on asi teisiti
- Arv kuvatakse märgina alles vastuvõtuseadmes)



```
sketch_nov02a
void setup() {
  // put your setup
  Serial.begin(9600);
  byte a;
  char b;
  a=65;
  Serial.println(a);
  b=a;
  Serial.println(b);
  b++;
  Serial.println(b);
  b++;
  Serial.println(b);
  b=36;
  Serial.println(b);
}
```

# Tekst (stringid)

- String on märkide jada, olemuselt ühebaidiste arvude massiiv.
- Saab teha kõiki tehteid.

```
sketch_nov02a | Arduino 1.6.7
File Edit Sketch Tools Help

sketch_nov02a
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  char a[13]={84,101,114,101,32,109,97,97,105,108,109,32,33} ;
  char b[]="Suvaline tekst";
  Serial.println(a);
  Serial.println(b);

  for (int i = 0; i < 13; i++){
    Serial.print(a[i]);
  }
}
```

```
Tere maailm !
Suvaline tekst
Tere maailm !
```

```
sketch_nov02a | Arduino 1.6.7
File Edit Sketch Tools Help

sketch_nov02a
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  byte a[13]={84,101,114,101,32,109,97,97,105,108,109,32,33} ;
  char b[]="Suvaline tekst";
  //Serial.println(a);
  Serial.println(b);

  for (int i = 0; i < 13; i++){
    Serial.print(a[i]);
  }
}
```

```
Suvaline tekst
841011141013210997971051081093233
```

# Tekst

- char vs byte

Kui on vaja numbrina defineeritud andmeid tekstiks muuta

```
sketch_nov02a | Arduino 1.6.7
File Edit Sketch Tools Help

sketch_nov02a
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  byte a[13]={84,101,114,101,32,109,97,97,105,108,109,32,33} ;
  char b[]="Suvaline tekst";
  //Serial.println(a);
  Serial.println(b);

  for (int i = 0; i < 13; i++){
    Serial.print(a[i]);
  }
}

COM38 (Arduino/Genuino Uno)
Suvaline tekst
841011141013210997971051081093233
```

```
sketch_nov02a | Arduino 1.6.7
File Edit Sketch Tools Help

sketch_nov02a
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  byte a[13]={84,101,114,101,32,109,97,97,105,108,109,32,33} ;
  char b[]="Suvaline tekst";
  //Serial.println(a);
  Serial.println(b);

  for (int i = 0; i < 13; i++){
    Serial.print(char(a[i]));
  }
}

COM38 (Arduino/Genuino Uno)
Suvaline tekst
Tere maailm !
```

# Programmi juhtimine (valiklause)

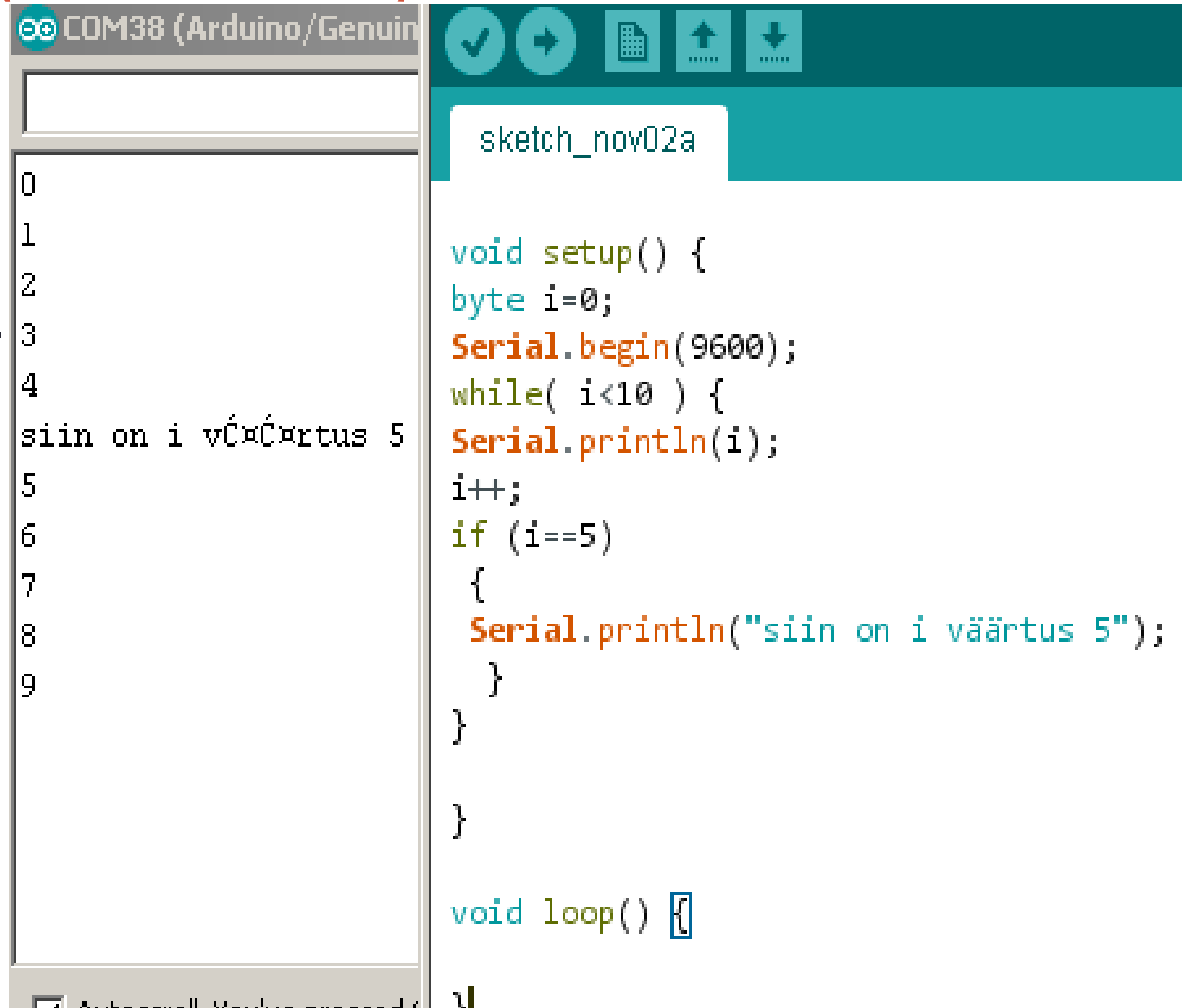
- **if** (tingimus)

lause1

**else**

lause 2

Kui lauset 2 ei ole vaja võib **else** ära jätta.



```
void setup() {  
  byte i=0;  
  Serial.begin(9600);  
  while( i<10 ) {  
    Serial.println(i);  
    i++;  
    if (i==5)  
    {  
      Serial.println("siin on i väärtus 5");  
    }  
  }  
}  
  
void loop() {
```

# Programmi juhtimine (valiklause)

- Pärast tingimust semikoolonit panna ei tohi , vastasel juhul täidetakse tühi lause ! C keele reeglite poolest on asi korras .

Eksivad ka kogenud programmeerijad

```
void setup() {  
  byte i=0;  
  Serial.begin(9600);  
  while( i<10 ) {  
    Serial.println(i);  
    i++;  
    if (i==5);  
    {  
      Serial.println("siin on i väärtus 5");  
    }  
  }  
}  
  
void loop() {  
  
}
```

Ma ju teen kõik õigesti aga.....

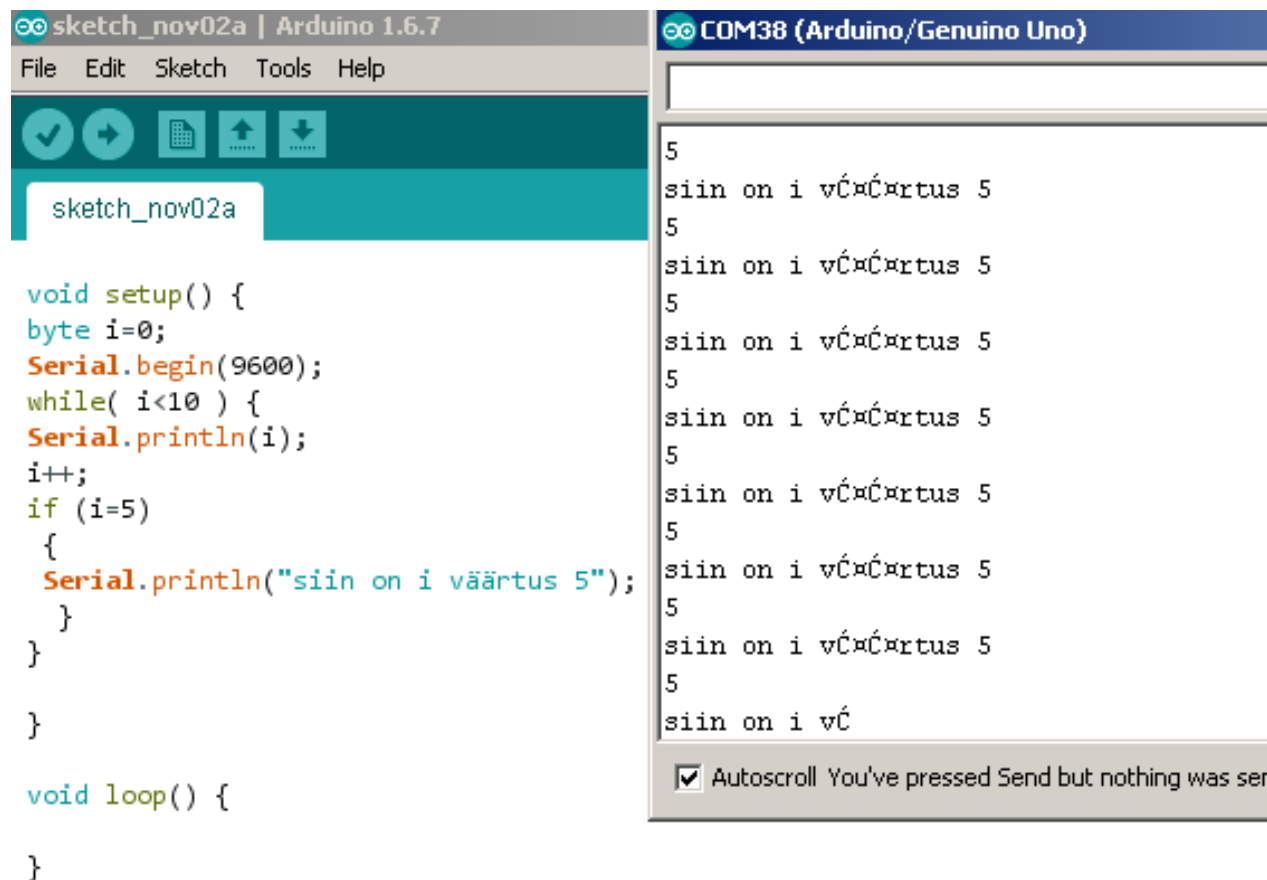
Arvuti ei täida meie soove ,vaid meie käske !

# Programmi juhtimine (valiklause)

- Teine tüüpiline loogikaviga (ka kogenud inimestel): == asemele pannakse =
- Arvuti teeb, mis kästakse, i=5 ja tehte tulemus on tõene...

Vähegi pikemast koodist (eriti kui selle teinud on vöõras - "aga miks ei tööta, tegin taas kõik õigesti ") seda üles leida on **väga raske** !

Mitte segi ajada = ja ==  
= on omistamine !

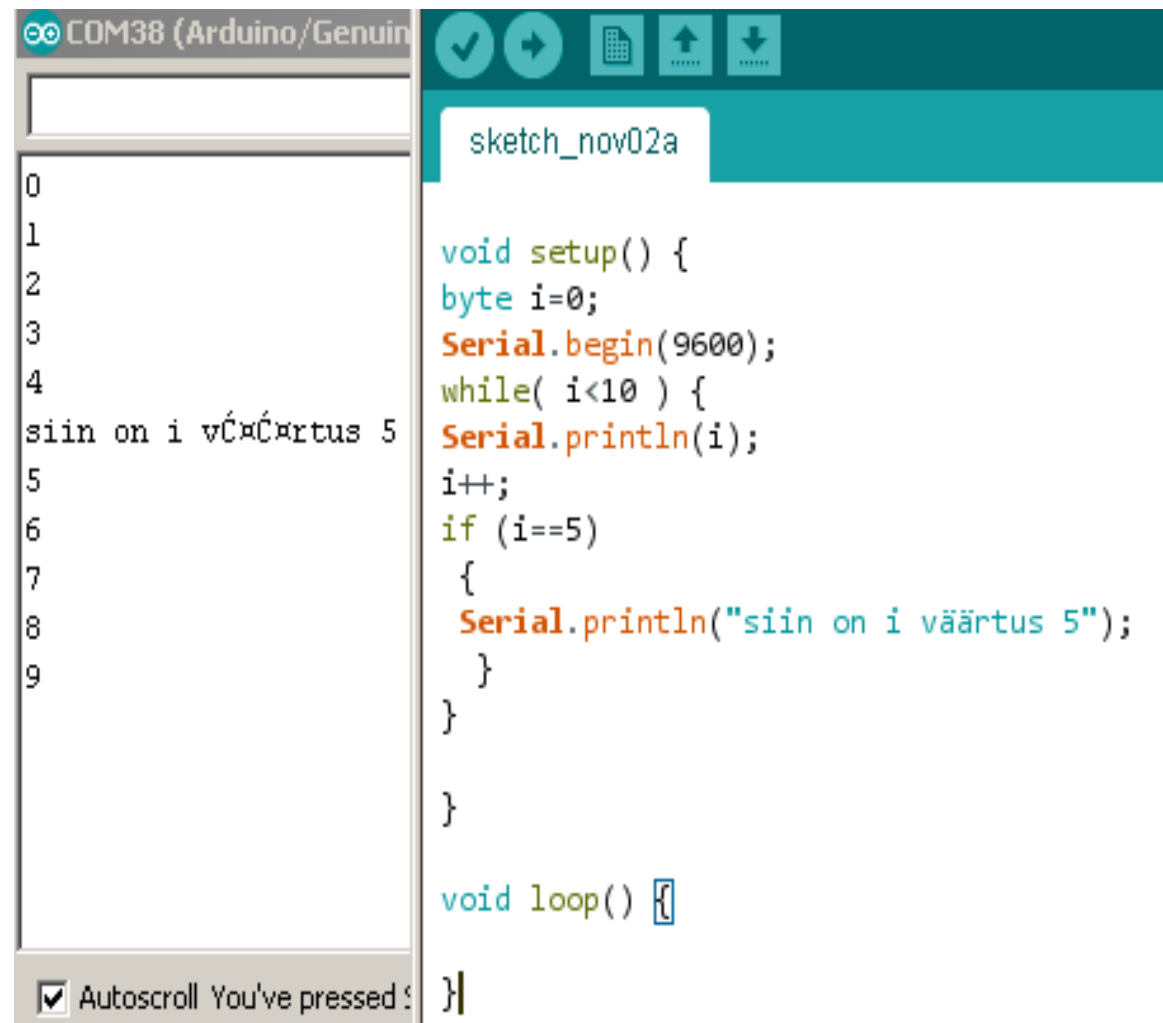
The screenshot shows the Arduino IDE interface. The main window displays a sketch named 'sketch\_nov02a' with the following code:

```
void setup() {  
  byte i=0;  
  Serial.begin(9600);  
  while( i<10 ) {  
    Serial.println(i);  
    i++;  
    if (i=5)  
    {  
      Serial.println("siin on i väärtus 5");  
    }  
  }  
}  
  
void loop() {  
  
}
```

The IDE indicates a compilation error: "Error: 'i' is not a constant expression." The serial monitor on the right shows the output of the program. It prints the numbers 0 through 9, with the number 5 appearing twice. The final output is 'siin on i v' followed by a carriage return and line feed. A status bar at the bottom of the serial monitor says 'Autoscroll You've pressed Send but nothing was ser'.

# Programmi juhtimine (kordus)

- **while** (tingimus) lause
- Või **do** lause **while** (tingimus)
- Täidetakse terve lause niikaua (korduvalt), kuni **while** tingimus on tõene.
- Tingimuste korral samad probleemkohad, mis if lauses (**=** vs **==** ) !
- Tsükli saab katkestada käsuga **break**;
- **break** katkestab vaid selle tsükli, mille sisse see kirjutatud on !



The screenshot shows the Arduino IDE interface. The top bar indicates the board is COM38 (Arduino/Genui...). The sketch name is 'sketch\_nov02a'. The code in the editor is as follows:

```
void setup() {  
  byte i=0;  
  Serial.begin(9600);  
  while( i<10 ) {  
    Serial.println(i);  
    i++;  
    if (i==5)  
    {  
      Serial.println("siin on i väärtus 5");  
    }  
  }  
}  
  
void loop() {
```

The serial monitor on the left shows the output of the program, with line numbers 0 through 9. The output is:

```
0  
1  
2  
3  
4  
siin on i väärtus 5  
5  
6  
7  
8  
9
```

At the bottom of the serial monitor, there is a checkbox for 'Autoscroll' which is checked, and the text 'You've pressed S'.

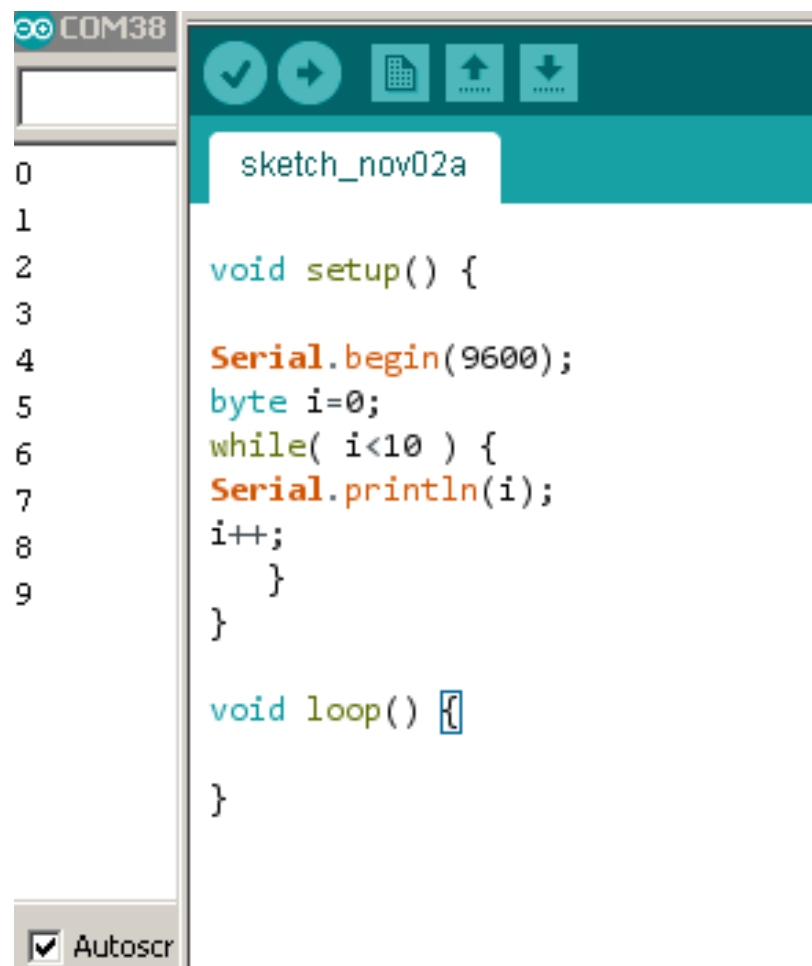


# Programmi juhtimine (loendus)

- **for** (avaldis1;avaldis2;avaldis3) lause
- Laused võivad ka puududa kuid semikoolonid peavad olema. Avaldis 3 järele semikoolonit ei panda !
- Kõigepealt täidetakse ettevalmistatav avaldis1, siis täidetakse lauset niikaua kuni avaldis2 annab mittenullise tulemuse, siis täidetakse avaldis3.
- Olemuselt teeb sama asja „while“
- avaldis1 while (avaldis2) {lause avaldis3}

# Programmi juhtimine (loendus)

- Tulemus on sama.
- while



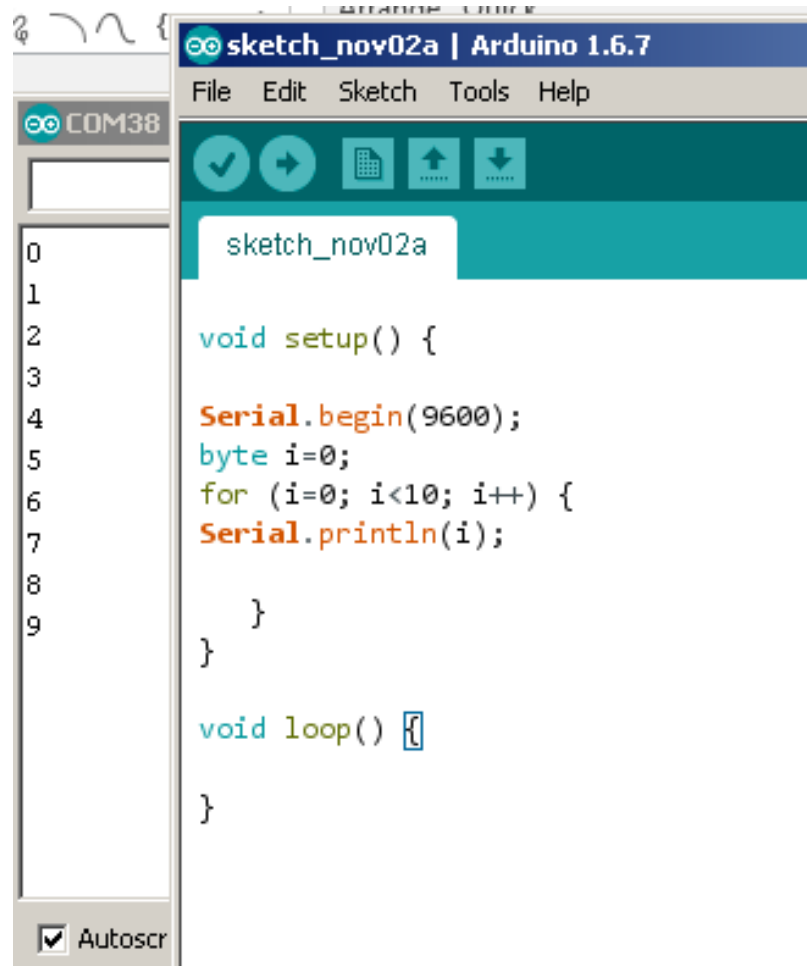
```
COM38
sketch_nov02a

0
1
2 void setup() {
3
4   Serial.begin(9600);
5   byte i=0;
6   while( i<10 ) {
7     Serial.println(i);
8     i++;
9   }
}

void loop() {
}
```

Autoscr

for



```
sketch_nov02a | Arduino 1.6.7
File Edit Sketch Tools Help
COM38
sketch_nov02a

0
1
2 void setup() {
3
4   Serial.begin(9600);
5   byte i=0;
6   for (i=0; i<10; i++) {
7     Serial.println(i);
8   }
9 }

void loop() {
}
```

Autoscr

# Programmi juhtimine (loendus)

- Pärast for tsükli kirjeldust semiloolonit panna ei tohi , vastasel juhul täidetakse vaid tühi lause ! C keele reeglite poolest on asi korras .
- Pärast tsükli täitmist on i väärtus 10, mis on korrektne , aga arvatavasti programmeerijale ebameeldiv .

The image displays three screenshots of the Arduino IDE interface, illustrating the execution of a program. The central screenshot shows the code editor with the following code:

```
void setup() {  
  Serial.begin(9600);  
  byte i=0;  
  for (i=0; i<10; i++){  
    Serial.println(i);  
  }  
}  
  
void loop() {  
}
```

The left screenshot shows the serial monitor output, which contains the number 10. The right screenshot shows the serial monitor output, which is empty. An arrow points from the text 'Pärast tsükli täitmist on i väärtus 10, mis on korrektne , aga arvatavasti programmeerijale ebameeldiv .' to the '10' in the serial monitor output of the left screenshot.

# Definitsioonid

- Saab lihtsustada programmeerija tööd, saab teha koodi loetavamaks, muudetavamaks, erineva riistvara jaoks.

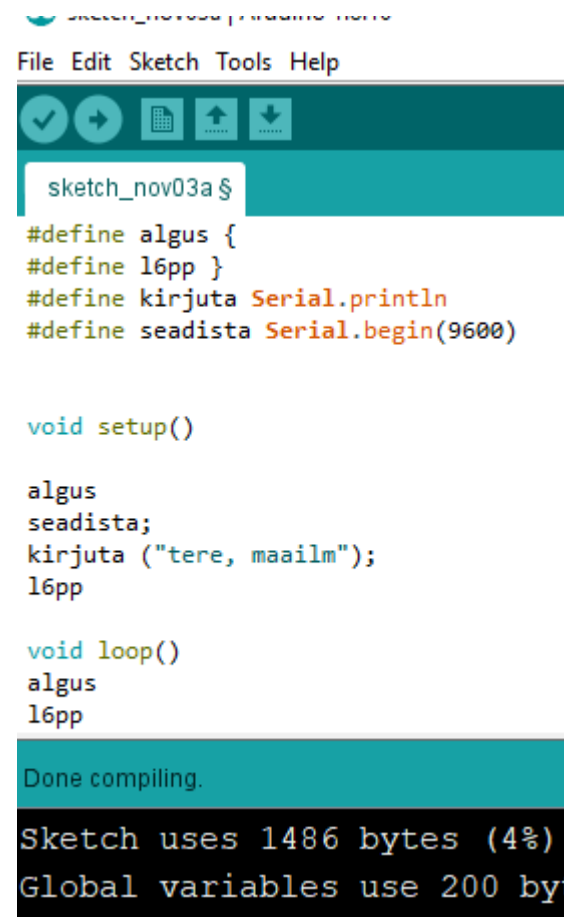
```
#define kartuleid 5 // igale poole kus on tekst "kartuleid" pannakse 5
```

```
#define porgandeid 12
```

```
#define doktorikraadiga_programmeerija Martin
```

- byte tulemus=kartuleid+porgandeid;
- byte tulemus=5+12; //sama mis eelmine rida
- Mugav, kui on vaja kasutada programmi mitmes kohas, lähteandmeid muutes.
- **Definitsioon ei ole muutuja !**
- **Ehk kartuleid=kartuleid+3 ;**
- **on vigane lause (omistamistehe !)**

Arduino "keele mugavused" on ka suures osas definitsioonid.



```
File Edit Sketch Tools Help
sketch_nov03a $
#define algus {
#define lõpp }
#define kirjuta Serial.println
#define seadista Serial.begin(9600)

void setup()

  algus
  seadista;
  kirjuta ("tere, maailm");
  lõpp

void loop()
  algus
  lõpp
```

Done compiling.

Sketch uses 1486 bytes (4%)  
Global variables use 200 by



Saab ka sedasi – Üks võimalikke viise, kuidas koodist arusaamine võõrastel äärmiselt ebamugavaks teha.

# Funktsioon

- Oma olemuselt lausete kogum, millele on antud indentifikaator (nimi) ning sisendparameetrid
- Näide:

```
int MinuFuktsioon ( int sisend1, int sisend 2)
```

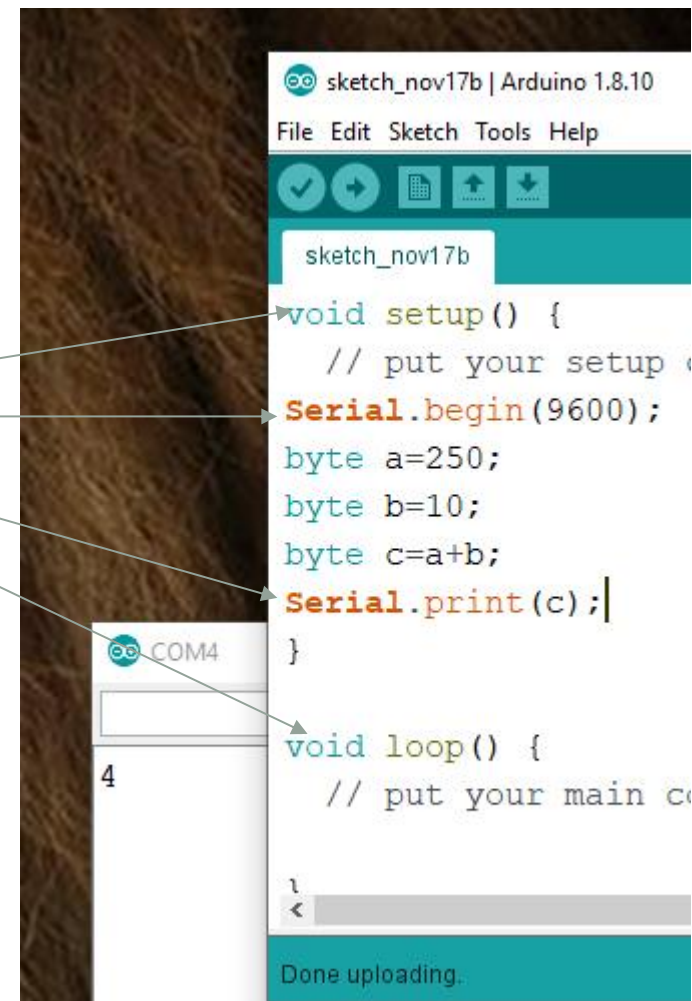
```
{  
  Int c=a+b;  
  return c;  
}
```

funktsioonid

Tagastatav väärtus määratakse käsuga return ja saab olla vaid üks.  
Kui funktsioon midagi ei tagasta (või pole sisendit), pannakse tüübiks **void**

C reeglitele vastav funktsioon **void teemidagi (void) { }**

Enamus Arduino „lisasid“ on teekidesse kirjutatud valmisfunktsioonid (**oranži värvi**) .....**pinMode...digitalWrite...analogRead** .....



# Näide - analogRead

- Kasutaja kirjutab `a=analogRead(A0);`
- Aga kutsutakse välja tegelikult see:
- C:\Program Files (x86)\Arduino\hardware\arduino\avr\cores\arduino\wiring\_analog.c
- See on vaid koodi algus.

```
38 int analogRead(uint8_t pin)
39 {
40     uint8_t low, high;
41
42     #if defined(analogPinToChannel)
43     #if defined(__AVR_ATmega32U4__)
44         if (pin >= 18) pin -= 18; // allow for channel or pin numbers
45     #endif
46     pin = analogPinToChannel(pin);
47     #elif defined(__AVR_ATmega1280__) || defined(__AVR_ATmega2560__)
48         if (pin >= 54) pin -= 54; // allow for channel or pin numbers
49     #elif defined(__AVR_ATmega32U4__)
50         if (pin >= 18) pin -= 18; // allow for channel or pin numbers
51     #elif defined(__AVR_ATmega1284__) || defined(__AVR_ATmega1284P__) ||
52         if (pin >= 24) pin -= 24; // allow for channel or pin numbers
53     #else
54         if (pin >= 14) pin -= 14; // allow for channel or pin numbers
55     #endif
56
57     #if defined(ADCSRB) && defined(MUX5)
58         // the MUX5 bit of ADCSRB selects whether we're reading from char
59         // 0 to 7 (MUX5 low) or 8 to 15 (MUX5 high).
60         ADCSRB = (ADCSRB & ~(1 << MUX5)) | (((pin >> 3) & 0x01) << MUX5);
61     #endif
```

