**printf (´´text´´)** – výpis textu, ktorý tam mam napísaný

**scanf** – vypísanie hodnoty int

**%d** – používame na výpis celého čísla rámci funkcie printf

**int** ´názov premeny´ .... deklarácia

**&&** - funguje ako ,,and,, v pythone

**||** - logicky OR

**cyklus for** – vypisuje čísla

**cyklus while** – podobny ako for, for vieme prerobiť na while

while (i < 5) .... podmienka

printf(“Ahoj/n“); <- vykoná sa niekoľkokrát ... while

i++ ... zväčšovanie premennej – potrebne

printf(“Ahoj/n“); <- vykoná sa raz ... do-while

i++ ... zväčšovanie premennej – potrebne

while (i < 5) .... podmienka na konci

goto begin – cyklus, po vykonaný sa vždy vráti naspäť na začiatok alebo niekde na začiatku – „skákanie v kóde“

v if keď je nule **if (0)** tak je to **VZDY NEPRAVDA – nevykoná** sa to ak je tam *niečo iné* if (-1) ... tak to berie za *pravdu – vykoná sa to*

**#include <stdlib.h>** // aby som mohol pouzit funkciu **exit**

**#include <stdlib.h>** // aby som mohol pouzit funkciu **atoi**

**/n** – novy riadok

**//t** - tabulátor

**//** - zobrazí

/ - biele znaky

**if(){}** - funkcia ktorej musíme dát určitú podmienku aby fungovala napr. if(num1 < num2){}

**sizeof()** –

**numsize** - ako ,,len,, čo napíše súčet všetkých prvkov v poli, napr. ak mam pole [1, 2, 3, 5, 10, 8] tak numsize je 6

**scanf("%d", &testInteger) -** čo napíšem tak to sa da do testInteger, predtým je väčšinou nejaká sprava oznamujúca že mate niečo pridať

**scanf** –načítanie, číta - - STDIN

**printf** – vypísanie - - STDOUT

**konštanty**: nemôžeme meniť, raz sa nastavi a hodnota tam ostava, nejaky typ (int, float)

const intConst1 = 10 .... definovane ako int, definícia a zároveň deklarácia, konštantná premenná

**premene**: možeme menit

int testInteger - vyhlásim že existuje nejaká premenná

**postfix** - za premenou (a++), robí to že ku číslu pripočíta niečo ale vypíše to až na druhýkrát ako to zavoláme

**prefix** – pred premenou (++a), robí to že počas použitia printf sa hneď aj pripočíta daná hodnota a aj sa výpiše výsledok

**možu byt aj minus (a--, --a)**

int a = 10 int a = 10

printf(a++) - 10 printf(++a) - 11

printf(a) – 11 printf(a) - 11

i++ - i zvýšime aby program nebežal donekonečna

**char –% s - reťazec**

**int – %d – celé číslo**

**float – %f – desatinne číslo**

**nesting –** vnorene

**selekcia –** vetvenie

**sekvencia –** postupnost –nieco ide zaradom

**cyklus -**

**= priradzujem do premennej**

**== keď porovnávam**

>= alebo <=

**Ked chceme pracovať s argumentami:**

**argc -** je premenná ktorá obsahuje počet argumentov vrátane programu

**argv** - je premenná ktorá obsahuje jednotlivé argumenty, počíta aj program (0 je program)

int num[] = (1,2,3,4,) ...pole a chceme vediet jeho velkost pouzijeme – **sizeof(num) / sizeof(int)**

**int var1 = 10 ... je definovaný takže je deklarovaný**

**var1 = 10**

**var2 = 20 ..... nie je definovane**

// SUMMARY

// **deklaracia vs definicia**

// **deklaracia** nenastavuje premennej hodnotu, len povie ze premenna bude neskor v kode existovat

// **definicia** priradi premennej hodnotu pocas jej vytvorenia (a tym zaroven deklaruje premennu)

// definicia nastavuje premennej nejaku hodnotu a hovori ze premenna zacina existovat

// je mozne deklarovat premennu bez definicie

// nieje mozne definovat premennu bez deklaracie

// definicia = deklaracia + priradenie

// v pythone sa pri definovani premennej neuvadza jej typ (premenna vznika priradenim)

// v jazyku C typ premennej uvadzam len pocas deklaracie alebo definicie

// pocas priradenia neuvadzam typ premennej

**main** - sa moze vyskytovat len 1x v programe - oznacuje sa niekedy ako entry point. je to funkcia ktora sa spusti ako prva ked zapneme program

**return 0** - potrebná vec na konci programu c

UBUNTU: **gcc argument.c**

**./a.out**

**./skompilovany.out**

**nested – vnorené**