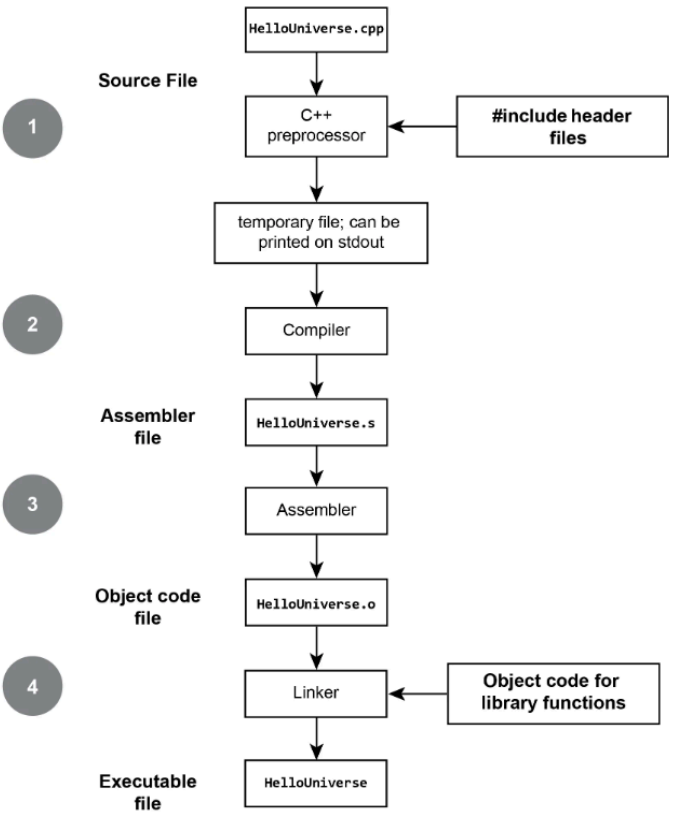
****

Tema 1

****

OSNOVNE INFORMACIJE O JEZIKU

- Istorijski
  - Produzetak jezika C, postoje 4 veca update c++11, c++14, c++17, c++20 i svaki ima neke specificne novitete
  - Nudi veliku mogucnost da programer upravlja memorijom i resursima
  - Razlika u odnosu na C je podrzavanje klasa i objekata, koje C nema
- Za sta se koristi
  - Dosta je brzi od Pythona
  - Moze da se koristi za pisanje operativnih sistema ili programa, ali i za pravljenje aplikativnog softvera
  - C ne koristi OOP paradigmu ali koristi neke druge bitne koncepte u programiranju koje cemo koristiti u Javi ili C++.
- Koje okruzenje koristimo
  - Moze da se koristi bilo koji tekst editor ali da odvojeno imamo instaliran kompajler za C++
  - Mi cemo koristiti CodeBlocks (u kome mozemo i C i C++ da kuckamo)
  - http://www.codeblocks.org/. Download mingw-setup.exe i instalirati (tekst editor i kompajler)
- Razlike - Python i C++ **(samo okvirno spominjati dok se rade primeri, ne pricato ovako na 'suvo')**
  - Više se potencira na razlicitim tipovima podataka
  - C++ ignorise spejseve
  - Funkcionalno programiranje kao i Pythonu - pri pokretanju programa, pokrece se main() funkcija
  - Sintaksna pravila su drugacija
  - Nemamo dvotacke za blokove nego viticaste zagrade
  - Svaka naredba se zavrsava sa tacka zarezom
  - Ostale naredbe cemo u toku rada spominjati i uporedjivati sa Pythonom
  - Komentari su umesto # naznaceni sa // za jedan red. Komentari za vise redova su /* i */
  - Buildovanje fajla kao proces



        - Nakon napisanog koda, kod se BUILDUJE nakon čega se
            - Ukljucuje C++ predprocesor (on obavlja sve include-ove)
            - Aktivira se kompajler i kao rezultat imamo asembler fajl
            - Aktivira se asembler i pravi objektni fajl
            - Aktivira se linker i on od objektnog, ukljucujuci razne biblioteke, pravi exe fajl
        - Nakon toga se program RUN-uje, cime se pokrece exe fajl
        - Obj i exe nisu postojali kod Pythona
        - U ovom slucaju sada imamo prevodjenje na masinski jezik (asembler)
- Tipovi podataka i problemi sa njima
    - Velicine u bajtovima, predstavljanje broja u binarnom zapisu
    - Overflow pojam
    - Sizeof funkcija
    - Signed vs unsigned

INSTALACIJA

- Potrebno je voditi učenika kroz instalacioni proces kod njega na racunaru (online nastava)
- Potrebno samo opisati proces instalacije ucenicima (nastava u ucionici)

PRIMER 1:

OBJAŠNJENJA:
```
#include <iostream>
using namespace std;

int main() {
     // na ekran ispisujemo pozdravnu poruku
     cout << ”Zdravo svete!” << endl;
     return 0;
}
```

- cout je console output
- Operator ispisa <<
- Operator upisa >>
- Svaki program mora biti napisan kroz funkciju main() koja se uvek pokrece
- Sa return 0 se signalizira da je rezultat izvrsavanja uspesan
- Telo funkcije ima neki tok, kao i svaka funkcija
- Inlude iostream omogucava da koristimo cout i endl
- Using namespace omogucava da prilikom koristenja cout i endl naredbi ne moramo da pisemo ime namespacea std::cout << ”Zdravo svete!” << std::endl;
- Izgled ovog programa u pythonu je po defaultu samo jedna linija `print(“Zdravo svete”)`
- Ako dodamo jos jedan cout sa porukom “Ucimo cpp”, a izbacimo endl; onda ce sve biti u istom redu, iako se naredbe napisu u dva odvojena reda.

- BITNO: napraviti paralelu sa naredbama iz drugih jezika koje ucenik zna (cout:print:printf, ili cin:input:scanf)

PRIMER 2: Input i output i promenljive / MOZE DA SE NASTAVI SAMO TAJ GORE PRIMER!

```
#include <iostream>
using namespace std;

int main() {
     cout << ”Kako se zoves?” << endl;
     string ime;
     cin >> ime;
     cout << ”Zdravo, ti se zoves ” << ime << endl;
     return 0;
}
```

OBJAŠNJENJA:

- C++ je staticki tipiziran jezik - za svaku promenljivu se unapred zna kojeg je tipa (kao kod C, a nismo imali u Pythonu)
- String je jedan od tipova podataka koje razlikujemo u c++, to je tekstualni tip podataka
- Deklaracija promenljive je proces kojom se definise tip neke promenljive
- Definicija promenljive je kada ona ima i neku pocetnu vrednost
- Vrednost promenljive se menja prilikom dodele
- Cin je kao input u pythonu
- Cin je skraceno od console input
- Primetiti da se spejs nalazi pre kraja stringa za ispis a pre imena, kako bi ime bilo odvojeno od teksta

PRIMER 3: Input ostalih tipova podataka - ne mora svih - samo njih par, int float, double

```
#include <iostream>
using namespace std;

int main() {
    cout << "Unesi integer" << endl;
    int c_broj;
    cin >> c_broj;
    cout << "Unesi realan broj" << endl;
    float f_broj;
    cin >> f_broj;
    cout << "Unesi integer" << endl;
    long long int l_broj;
    cin >> l_broj;
    cout << "Unesi realan broj" << endl;
    double d_broj;
    cin >> d_broj;
```

```
    cout << "Unesi integer" << endl;
    short s_broj;
    cin >> s_broj;
    cout << "Unesi jedno slovo" << endl;
    char slovo;
    cin >> slovo;
    cout << c_broj << " " <<sizeof(c_broj) << endl;
    cout << f_broj << " " <<sizeof(f_broj) << endl;
    cout << l_broj << " " <<sizeof(l_broj) << endl;
    cout << d_broj << " " <<sizeof(d_broj) << endl;
    cout << s_broj << " " <<sizeof(s_broj) << endl;
    cout << slovo << " " <<sizeof(slovo) << endl;
    // razlicito su od sistema do sistema velicine ovih tipova podataka
    short a = 32766;
    short b = 4;
    short c = a+b;
    cout << c; // neocekivano ponasanje, netacan zbir

    return 0;
}
```

OBJAŠNENJE:

- **Char 1B, bool 1B, short 2B, int 4B, float 4B, long long int 8B, double 8B**
- U slucaju da se ne koristi pravi tip podatka za smestanje nekog podatka, ocekuju se greske u racunanju ili greske u radu samog programa
- Sta znaci da je potrebno 2B za short?
    - 2B = 16 bitova
    - 2^15 do 2^0 su bitovi za short
    - Svaki od datih brojcanih tipova podataka moze da bude signed i unsigned, po default se podrazumeva da je signed, to se ne pise
    - Signed je po default, kako se ne navede, podrazumeva se (to znaci da su podrzani i pozitivni i negativni brojevi)
    - Unsigned znaci da se koriste samo nenegativni brojevi
- Ako imamo short a = 1 i int a = 1, to su razlicite reprezentacije bitova i razlicito zauzece memorije
    - 00000000 00000001
    - 00000000 00000000 00000000 00000001
- Ukoliko imamo int + double, on ce se cuvati u double
- **Ovo je samo okvirno potrebno objasniti, ne ulaziti u detalje!**

Binary value	Sign–magnitude interpretation	Unsigned interpretation
00000000	0	0
00000001	1	1
⋮	⋮	⋮
01111101	125	125
01111110	126	126
01111111	127	127
10000000	−0	128
10000001	−1	129
10000010	−2	130
⋮	⋮	⋮
11111101	−125	253
11111110	−126	254
11111111	−127	255