

Korišćene oznake:

- kada se navodi sadržaj nekog fajla, sadržaj će biti ograden linijama (linije nisu dio fajla):

<sadržaj fajla>

- tri tačke (...) u jednoj liniji sadržaja nekog fajla označavaju da je više praznih linija izbačeno.

Sistemske promjenljive

Sistemske promjenljive su promjenljive koje su za potrebe korišćenja u komandnoj liniji definisane od strane operativnog sistema. Najčešće sadrže tekst. Jedna sistemska promjenljiva je promjenljiva "PATH" koja sadrži niz direktorijuma u kojima se prilikom pokretanja programa iz komandne linije podrazumjevano traže programi. Po potrebi je moguće definisati i nove sistemske promjenljive, kao i izmjeniti sadržaj postojećih linija. U Windows-u se to radi tako što se desnim tasterom klikne na Computer (ili My Computer u verziji XP), zatim lijevom klikne na Properties, zatim klikne na Advanced system settings (tab Advanced u verziji XP) i na kraju klikne na Environment Variables. U prozoru koji se otvori je moguće dodati novu promjenljivu klikom na New ili izmjeniti postojeću odabirom odgovarajuće i klikom na Edit. Svaka promjenljiva ima svoj naziv i vrijednost. Ukoliko je to moguće, sve izmjene praviti nad promjenljivim za tekućeg korisnika. Za potrebe daljeg rada ovog projekta je potrebno u promjenljivu PATH dodati i putanju do bin foldera unutar foldera bc31. Ako je bc31 folder snimljen u root C drajva, onda je prilikom pokretanja kompajlera dovoljno pisati "bcc" umjesto "C:\bc31\bin\bcc". Prilikom dodavanja nove putanje, na postojeću vrijednost promjenljive PATH treba dopisati prvo dvotačku, a zatim putanju zaključno sa bin folderom (npr. C:\bc31\bin). U komandama se promjenljive koriste tako što se između dva znaka % navede naziv promjenljive (npr. %PATH%). Ovo je korisno ukoliko se definišu sistemske promjenljive lib i include koje sadrže putanje do korišćenih biblioteka i fajlova zaglavlja.

Objašnjenje fajlova sa ekstenzijom ".bat".

Ovi fajlovi su izvršni fajlovi koji u obliku teksta sadrže niz komandi koje se pokreću pokretanjem skripta. ".bat" fajlovima je moguće proslijediti argumente koji se mogu koristiti u okviru komandi navedenih u fajlu. Parametrima se pristupa po rednom broju (za pristup prvom argumentu se koristi "%1" – bez navodnika). U komandama unutar ".bat" fajlova je moguće koristiti i sistemske promjenljive, tako što se naziv promjenljive navede između dvostrukih znakova % (%%naziv%%)

U prilogu su data tri ".bat" fajla:

- "cl.bat" – sadrži komandu za prevodjenje jednog fajla (fajl se prosledjuje kao parametar skripta). Primjer pokretanja:
cl z5n.cpp
- "cva.bat" – sadrži komandu za generisanje asemblerskog izlaza umjesto izvršnog programa (ulaz je samo jedan fajl)
- "preproc.bat" – sadrži komandu za obradu jednog ulaznog programa od strane preprocesora

Preprocesor

Obrađuje sve direktive u ulaznim fajlovima. Najčešće korišćene direktive:

- `#include "fajl"` – direktivu zamjenjuje sadržajem navedenog fajla; fajl traži i u lokalnom direktorijumu;
- `#include <fajl>` – direktivu zamjenjuje sadržajem navedenog fajla; fajl ne traži u lokalnom direktorijumu;
- `#define simbol sadržaj` – definiše navedeni simbol i dodijeljuje mu zadati sadržaj;
- `#ifndef simblo` – početak bloka koji se prepisuje u izlazni fajl samo ukoliko dati simbol nije definisan;
- `#endif` – kraj `#ifndef` direktive.

Primjeri upotrebe:

- `zagl1.h` za potrebe zaštite od višestrukog uključivanja:
"zagl1.h":

`#ifndef _ZAGL1_H_`
`#define _ZAGL1_H_`

`#include "zagl2.h"`

`class A{`
`public:`
 `A();`
`};`

`#endif`

"zagl2.h":

`#ifndef _ZAGL2_H_`
`#define _ZAGL2_H_`

`#include "zagl1.h"`

`class B {`
`public:`
 `B();`
`};`

`#endif`

"main.cpp"

```
#include <zagl1.h>
#include <zagl2.h>
```

```
A::A(){
    //
}
```

```
B::B(){
    //
}
```

```
int main(){
    return 0;
}
```

Generisani izlaz u "main.i":

```
main.cpp 1:
.\zagl1.h 1:
...
.\zagl1.h 4:
zagl2.h 1:
...
zagl2.h 4:
zagl1.h 1:
...
zagl1.h 12:
zagl2.h 5:
zagl2.h 6: class B {
zagl2.h 7: public:
zagl2.h 8: B();
zagl2.h 9: };
...
zagl2.h 12:
.\zagl1.h 5:
.\zagl1.h 6: class A{
.\zagl1.h 7: public:
.\zagl1.h 8: A();
.\zagl1.h 9: };
...
.\zagl1.h 12:
main.cpp 2:
.\zagl2.h 1:
...
.\zagl2.h 12:
main.cpp 3:
main.cpp 4: A::A(){
```

komentari (nisu dio izlaza)

na pocetku main.cpp ukljucen zagl1.h
nije definisano _ZAGL1_H_
definise se _ZAGL1_H_
ukljucuje se zagl2.h (iz zagl1.h)
nije definisano _ZAGL2_H_
definise se _ZAGL2_H_
ukljucuje se zagl1.h (iz zagl2.h)
definisano _ZAGL1_H_ i zato
se ostatak fajla zagl1.h
preskače
završeno uključivanje zagl1.h iz zagl2.h
ostatak sadržaja zagl2.h

završeno uključivanje zagl2.h iz zagl1.h
ostatak sadežaja zagl1.h

završeno uključ. zagl1.h iz main.cpp
uključuje se zagl2.h iz main.cpp
ali je sada definisano _ZAGL2_H_
pa se ostatak zagl2.h preskače
ostatak sadržaja main.cpp

```

main.cpp 5:
main.cpp 6: }
main.cpp 7:
main.cpp 8: B::B(){
main.cpp 9:
main.cpp 10: }
main.cpp 11:
main.cpp 12: int main(){
main.cpp 13: return 0;
main.cpp 14: }
main.cpp 15:

```

Zbog korišćenja <> u #include direktivi u fajlu "main.cpp", preprocesoru je neophodno u listi direktorijuma u kojima se traže fajlovi zaglavlja dodati i direktorijum koji sadrži fajlove "zagl1.h" i "zagl2.h". Ako se <> zamjeni sa "", to nije potrebno.

- Definisanje parametrizovanih makroa
"makroi.cpp"

```

#define naziv_makroa(param1, param2) param1param2 param1 param2 param1##param2 \
    param1##nastavak

```

```

naziv_makroa(argument1, argument2)

```

```

#define makro1 telo1
#define makro2 telo2

```

```

naziv_makroa(makro1, makro2)

```

```

#define novi_makro(p1,p2) naziv_makroa(p1, p2)

```

```

novi_makro(makro1,makro2)

```

```

#define makro3(param) param 1

```

```

neki tekst makro3(makro3(makro)) ostatak teksta

```

```

#define makro3(param) param##1

```

```

neki tekst makro3(makro) ostatak teksta

```

"makroi.i":

```

makroi.cpp 1:
makroi.cpp 2:
makroi.cpp 3:
makroi.cpp 4: param1param2 argument1 argument2 argument1argument2
    rgument1nastavak
makroi.cpp 5:
makroi.cpp 6:
makroi.cpp 7:

```

```

makroi.cpp 8:
makroi.cpp 9: param1param2 telo1 telo2 makro1makro2 makro1nastavak
makroi.cpp 10:
makroi.cpp 11:
makroi.cpp 12:
makroi.cpp 13: param1param2 telo1 telo2 telo1telo2 telo1nastavak
makroi.cpp 14:
makroi.cpp 15:
makroi.cpp 16:
makroi.cpp 17: neki tekst makro 1 1 ostatak teksta
makroi.cpp 18:
makroi.cpp 19:
makroi.cpp 20:
makroi.cpp 21: neki tekst telo1 ostatak teksta
makroi.cpp 22:

```

Proces zamjene poziva nekog makroa odgovarajućim sadržajem naziva se (makro)ekspanzija. Ukoliko su argumenti makroi, na mjestima gdje je korišćen parametar prosleđuje se ekspanzovani sadržaj argumenta, osim u slučaju upotrebe operatora ##, kada se prvo ugradi originalni argument (linija 4), pa se tek u ekspanzovanom sadržaju pokušava pronaći i ekspanzovati neki makro (linija 21). Poziv makroa mora imati jedan blanko znak ispred i jedan blanko znak iza. Ukoliko je definicija makroa predugačka, moguće je prelomiti makro u više linija, s tim da se na kraju svake linije osim poslednje doda znak '\' (ovaj znak mora biti neposredno ispred znaka za prelazak u novi red).

Prevođenje iz komandne linije

Proces dobijanja izvršnog programa od fajlova sa izvornim kodom programa zahtjeva dva koraka. Prvi korak je kompajliranje svih izvornih fajlova. Fajlovi se kompajliraju zasebno, bez obzira da li se kompajleru proslijede zajedno ili svaki pojedinačno. Kao rezultat kompajliranja dobijaju se objektni fajlovi koji se u drugom koraku povezuju (linkuju) u jedan izvršni fajl. Alati za prevođenje često omogućavaju da se pozove samo jedan program, a da on završi (sam ili korišćenjem drugih programa) i prevođenje i linkovanje. Tako se pozivanjem kompajlera bcc.exe sa:

```
bcc main.cpp
```

prevodi i linkuje fajl main.cpp u main.exe. Ukoliko treba prevesti i linkovati više fajlova, iza bcc se navodi lista fajlova odvojenih razmacima:

```
bcc fajl1.cpp fajl2.cpp
```

Prilikom prevođenja, ispred fajlova za prevođenje moguće je navesti i opcije. Opcije se navode tako što se navede znak '-' i potom naziv opcije, i ako opcija ima parametre, navode se i argumenti opcije. Kompletan spisak je moguće pronaći u dokumentaciji uz BCC sa sledeće adrese:

http://www.bitsavers.org/pdf/borland/Borland_C++_Version_3.1_Users_Guide_1992.pdf

Neke od korisnih opcija:

- mh – huge memorijski model,
- mt – tiny memorijski model (korišćen u nekim primjerima gdje se kao adresa uzimao samo offset),
- l<putanje odvojene sa> - spisak putanja do svih direktorijuma u kojima se traže ".h" fajlovi,
- L<putanje odvojene sa> - spisak putanja do svih direktorijuma u kojima se traže biblioteke,
- c – želi se samo proces kompajliranja, bez linkovanja; posle kompajliranja treba upotrijebiti tlink,

-S – umjesto izvršnog programa se želi generisanje asemblerskih fajlova.

Opcije je moguće navesti u komandnoj liniji, ali i u posebnom tekstualnom fajlu (pogledati sadržaj "cva.bat" kao primjer prosleđivanja opcija u komandnoj liniji i, "cl.bat" i "cf.txt" zajedno kao primjer skupljanja svih opcija u konfiguracioni fajl). Napomena: za otvaranje ".bat" fajlova kliknuti desnim tasterom na fajl i odabrati opciju "Edit". Slično konfiguracionim fajlovima, moguće koristiti i response fajlove koji mogu da sadrže kompletan spisak opcija i fajlova (neophodno koristiti kada je spisak predugačak da se proslijedi kroz komandnu liniju).

Kada je zajedno sa programom potrebno povezati i biblioteku, moguće je nakon liste cpp fajlova navesti i naziv biblioteke (npr. applicat.lib).

VAŽNO: Dužina naziva fajla ne smije biti veća od 8 znakova.