

# Vad förväntas ni kunna - Kompilering

Vad gör en kompilator.

- Översätter från skriven kod till maskinkod.

Hur får man tillgång till kod skapad av andra.

- Genom importering/länkning till annans kod

Kunna ta reda på vilken funktionalitet en viss header-fil kan tillhandahålla.

## Kompilering:

Läromål: Kompilering sker i flera steg. Man kan använda färdiga programdelar genom include-sattser.

### Färgförklaring

**Gult** är läsbara filer som du som programmerare skapat.

**Blått** är program som gör ett visst steg i skapandet av det färdiga programmet.

**Grönt** är kod som skapats av en kompilator

**Rött** är det färdiga programmet som du kan leverera till din uppdragsgivare.

Källkodsfiler filnamn1.c, filnamn2.c,

## Fördefinierade programdelar...

...är något som man kan återanvända, exempelvis för kryptering så att man själv slipper skriva allt från grunden.

Att vi vill använda denna typ av färdiga programdelar anger vi genom att i vår fil exempelvis skriva:

```
#include <stdio.h>
```

Då inkluderar vi headerfiler som visar vad som finns tillgängligt.

## Preprocessorn:

Utför makron, alltså modifierar .c-filen utifrån vad man begärt. **#define MAX 8**

## Preprocessormodifierad fil.

## C-kompilatorn:

Gör källkod till assemblerkod

## Assemblerfiler: (Av kompilatorn skapta).

filnamn1.asm / filnamn1.s

## Assembler:

Skapar processor-specifik kod.

## Maskinkod: (Av kompilatorn skapta).

filnamn1.obj / filnamn1.o

**Länkare:** Sätter ihop flera olika filer med maskinkod till En körbar fil.

**Färdigt program:** filnamn.exe

# Kompilering.

Läromål: Kompilering är en successiv översättning från människospråk till maskinspråk.

## Från läsbart

(filnamn.c)

897 bytes

## Till assemblerkod

(filnamn.s)

1.418 bytes

## Till binärkod (objekt)

(filnamn.o)

2.582 bytes

(a.exe)

17.564 bytes

```
// För att kunna använda sc
#define _CRT_SECURE_NO_WARN
////////////////////////////////////

#include <stdio.h>
#include <stdlib.h>

int main(int antalArgs, cha
{
    float tal, sum, storsta
    int ant;
    sum = 0;
    storsta = 0;
    ant = 0;
    printf("Skriv in talen.
    printf("Avsluta med ENI
    while(scanf("%f", &tal)
    {
        ant++;
        sum += tal;
        if (tal > storsta)
            storsta = tal;
    }
    printf("Medelvärde:
    printf("Storsta: %f",
    return 0;
}
```

```
test.s - Notepad
File Edit Format View Help

.scl 2; .type 32
pushl %ebp movl %e
$72, %esp movl $L
_printf leal -40(%ebp),
(%esp) movl $LC1, (%es
leal -40(%ebp), %eax mo
movl $LC2, (%esp) ca
$LC3, (%esp) call _p
(%ebp), %eax movl %e
$LC4, (%esp) call _s
(%esp) call _printf le
movl %eax, 4(%esp) mo
call _scanf flds -4
-48(%ebp) fstpl 4(
(%esp) call _printf fl
fadds -48(%ebp) fl
%st, %st(1) fstpl 4(
(%esp) call _printf le
_charBychar .def _c
2; .type 32; .e
pushl %ebp movl %e
$40, %esp movl %e
$20, -8(%ebp) movl -8
```

```
test.o - Notepad
File Edit Format View Help


L J J J J J .text
. .data
@
A.bss
€ A.rdata
€ T,
@ @U%áfîHÇJ$ è
E0%D$ÇJ$ è E0%D
$ÇJ$ è ÇJ$! è
E0%D$ÇJ$ è ÇJ$8
è E0%D$ÇJ$ è
ÛE00E0Y\ $ÇJ$K è
ÛE00E0Ût PÙY\ $ÇJ$Z
è ÉÁU%áfî(%e0ÇE0q
<E0fAçfAçAçAç%Èè<Èè
D$ÇJ$AçAçAç%ÈiÇE0 ÇJ$
è è ^EY%îEY<
tB<E0;E0}:<E0%AçîEY<Mî~
E0Y ÇJ$ è ÇJ$Y%$è
ÇJ$ è
èè<e0ÉÁU%áfîçfäð,
```

```
MZ L J yY .
DOS mode.
$ PE L | •ü2R T
+ + P
A.rdata @ 0
@ A
%0<uü<]0%î]Aç = " At½=
öt^è0, »yyyyçJ$ç '
$ç@ èAç jç@@ ...Atxf+ @
%çè fäðç- <0%L$ç<+
yç0P@ èÉpyyç & U%áfî
%Èi<Eiè0 è-
0ÛE0ÛE0ÛE0ÛE0Bäzw,éÉ<EÜ%
ù@1@ rç]AU%â0â]AU%áfîç
è:pyy[[]Aç,,@ 1A...Éè<ç
[[]Aç,,@ 1A...Éè<ç,,@ ..
É Éyçf;<uç%0<]üÉA',,0ç
Étçe0[^_]ACE^AAAA;`0@
%ÈEiç0@ %Èiçx0@ %E0iç(
' ÇCçAçç jç@@ Çç<
ÉAç,,*Hyçyçç;`0@ %...f
HVVV%çVç-Pç ç.0fîç...VÜB
```

# Kompilering.

Läromål: Kunna ta reda på information om vilka färdigskrivna programdelar man kan använda från en specifik header-fil.

Vi ska titta på: <http://www.cplusplus.com/reference/cstdio/>



```
// För att kunna använda scanf
#define _CRT_SECURE_NO_WARNINGS
////////////////////////////////////

#include <stdio.h>
#include <stdlib.h>

int main(int antalArgs, char** args)
{
    float tal, sum, storsta;
    int ant;
    sum = 0;
    storsta = 0;
    ant = 0;
    printf("Skriv in talen.\n");
    printf("Avsluta med ENLIG\n");
    while(scanf("%f", &tal) != EOF)
    {
        ant++;
        sum += tal;
        if (tal > storsta)
            storsta = tal;
    }
    printf("Medelvärde: %.2f\n", sum/ant);
}
```

# Kompilatorväxlar:

- Styr hur kompilatorn skall arbeta och vilken information och output man får vid kompileringen.
- Kan ges via menyer i en IDE (Integrated Development Environment) eller via växlar (exempelvis -S).
- Växeln --help listar tillgängliga växlar.
- gcc --help
- Oftast görs alla stegen från källkod till körbar fil utan att vi behöver bry oss.
- Kompilatorsväxlar behöver vi vanligtvis inte bry oss om. Just nu behöver ni bara veta att de finns.

Läromål: Man kan påverka kompileringen med växlar.

```

cmd _cmd.exe
>gcc --help
Usage: gcc [options] file...
Options:
  -pass-exit-codes      Exit with highest error code from a phase
  --help                Display this information
  --target-help         Display target specific command line options
  (Use '-v --help' to display command line options of sub-processes)
  -dumpspecs            Display all of the built in spec strings
  -dumpversion           Display the version of the compiler
  -dumpmachine          Display the compiler's target processor
  -print-search-dirs     Display the directories in the compiler's sea

  -print-libgcc-file-name Display the name of the compiler's companion
  -print-file-name=<lib> Display the full path to library <lib>
  -print-prog-name=<prog> Display the full path to compiler component <
  -print-multi-directory Display the root directory for versions of li
  -print-multi-lib       Display the mapping between command line opti
                        multiple library search directories
  -print-multi-os-directory Display the relative path to OS libraries
  -Wa,<options>          Pass comma-separated <options> on to the asse
  -Wp,<options>          Pass comma-separated <options> on to the prep

  -Wl,<options>          Pass comma-separated <options> on to the link
  -Xassembler <arg>     Pass <arg> on to the assembler
  -Xpreprocessor <arg>  Pass <arg> on to the preprocessor
  -Xlinker <arg>         Pass <arg> on to the linker
  -save-temps            Do not delete intermediate files
  -pipe                 Use pipes rather than intermediate files
  -time                 Time the execution of each subprocess
  -specs=<file>          Override built-in specs with the contents of
  -std=<standard>        Assume that the input sources are for <standa
  -B <directory>        Add <directory> to the compiler's search path
  -b <machine>          Run gcc for target <machine>, if installed
  -U <version>          Run gcc version number <version>, if install
  -v                   Display the programs invoked by the compiler
  -###                 Like -v but options quoted and commands not e
  -E                   Preprocess only; do not compile, assemble or
  -S                   Compile only; do not assemble or link
  -c                   Compile and assemble, but do not link
  -o <file>             Place the output into <file>
  -x <language>         Specify the language of the following input f
                        Permissible languages include: c c++ assemble
                        'none' means revert to the default behavior o
                        guessing the language based on the file's ext
  
```

## Mer om kompilering?

Vill du läsa mer om hur kompilering går till?

- <http://www.geeksforgeeks.org/understanding-extern-keyword-in-c/>
- <http://www.lurklurk.org/linkers/linkers.html> (Ungefär första halvan).
- <http://www.tenouk.com/ModuleW.html>

(Prioritera läsanvisningar och övningsuppgifter).

# Vad förväntas ni kunna - Kompilering

Vad gör en kompilator.

- Översätter från skriven kod till maskinkod.

Hur får man tillgång till kod skapad av andra.

- Genom importering/länking till annans kod

Kunna ta reda på vilken funktionalitet en viss header-fil kan tillhandahålla.