

Laboratorium Informatika dan Komputer

# Pemrograman Dasar (C++)

DEPARTEMEN TEKNIK ELEKTRO DAN TEKNOLOGI INFORMASI

FAKULTAS TEKNIK UNIVERSITAS GADJAH MADA

YOGYAKARTA

2022

## KATA PENGANTAR

Puji syukur kehadiran Allah SWT atas limpahan rahmat, karunia, serta petunjuk-Nya sehingga modul praktikum Pemrograman Dasar telah terselesaikan dengan baik. Dalam penyusunan modul praktikum ini, tim penulis telah banyak mendapatkan arahan, bantuan, serta dukungan dari berbagai pihak. Modul praktikum ini disusun berdasarkan Kurikulum Program Studi Sarjana Teknologi Informasi, Fakultas Teknik, Universitas Gadjah Mada tahun 2021.

Semoga modul praktikum ini bisa dimanfaatkan dengan baik. Tidak ada gading yang tidak retak, modul ini tentu memiliki banyak kekurangan yang InsyaAllah akan diperbaiki seiring dengan berjalannya waktu,

Hormat kami,

Tim Penyusun

## DAFTAR ISI

KATA PENGANTAR	2
Daftar Isi	3
MODUL 0 : Petunjuk Keselamatan Praktikum	6
Setting environment	6
Pengenalan GCC	18
MODUL 1 : Tipe Data Dasar, Operator, dan Pointer	
MODUL 2 : Control Statement: Selection Structure	
MODUL 3 : Control Statement: Repetition	
MODUL 4 : Struktur Data	
MODUL 5 : Pemrograman modular	
MODUL 6 : aktivitas praktikum	

## PETUNJUK KESELAMATAN PRAKTIKUM

### PROSEDUR KESELAMATAN PRAKTIKUM

Dalam rangka memperbaiki kualitas praktikum maka sebelum praktikum dimulai asisten atau laboran menginformasikan hal-hal yang berkaitan dengan keselamatan mahasiswa praktikan di laboratorium. Hal-hal yang harus diinformasikan kepada mahasiswa praktikan antara lain:

1. Letak pintu keluar dan pintu keluar darurat
2. Himbauan untuk meletakkan barang di tempatnya
3. Himbauan untuk menggunakan alat bantu keselamatan
4. Himbauan untuk tidak bercanda yang berlebihan pada saat praktikum.

Aturan keselamatan harus dibaca mahasiswa sebelum memulai praktikum. Berikut ini adalah aturan keselamatan umum dan khusus. Aturan keselamatan umum berlaku umum di semua laboratorium dalam lingkungan Departemen Teknik Elektro dan Teknologi Informasi, sedangkan aturan keselamatan khusus berlaku khusus di laboratorium tertentu. Mahasiswa diwajibkan untuk mengikuti petunjuk keselamatan berikut dengan disiplin yang tinggi dan penuh tanggung jawab.

### PETUNJUK KESELAMATAN UMUM

Untuk meningkatkan kualitas keamanan dan kenyamanan dalam praktikum dan suasana akademis, serta *personal safety*, maka aturan-aturan berikut ini harus ditaati dengan tegas. Pelanggaran terhadap aturan berikut akan dikenakan sanksi akademis yang berupa tidak diperbolehkan mengikuti praktikum (ringan) sampai dengan di-skors selama satu semester (berat).

1. Mahasiswa diwajibkan meletakkan tas dan barang-barang pribadi di tempat yang telah disediakan.
2. Mahasiswa dilarang keras untuk membawa alat komunikasi berupa handphone atau sejenisnya yang bisa mengganggu konsentrasi dalam praktikum. Harus diingat bahwa praktikum yang Anda lakukan mengandung resiko keselamatan.
3. Mahasiswa yang berambut panjang, melebihi bahu harus merapikan rambutnya dengan cara diikat atau sejenisnya.
4. Semua personel yang ada di laboratorium dilarang merokok.
5. Mahasiswa diwajibkan untuk menggunakan:
  - a. celana panjang dan baju/kaos berkerah selama praktikum dan berpakaian rapi dan sopan bagi mahasiswa, serta berpakaian rapi dan sopan bagi mahasiswi.
  - b. Menggunakan sepatu dan bukan sandal atau sepatu sandal. Disarankan sepatu yang digunakan adalah sepatu yang berbahan karet atau isolator yang tidak menghantar.
6. Mahasiswa berkewajiban menggunakan peralatan keselamatan tambahan yang diwajibkan di laboratorium tertentu.
7. Mahasiswa wajib menjalankan praktikum berdasarkan buku petunjuk/modul praktikum. Mahasiswa harus bertanya kepada laboran atau asisten jika tidak memahami apa yang dilakukan selama praktikum.
8. Jika terjadi hal-hal yang luar biasa misalnya gempa bumi atau kebakaran, maka keselamatan mahasiswa praktikan lebih diutamakan. Mahasiswa harus mendengarkan keterangan dari laboran atau asisten atau petugas keselamatan yang ditunjuk oleh Departemen.

9. Mahasiswa diwajibkan untuk merapikan alat-alat praktikum, mengembalikan ke tempat semula dan mematikan alat-alat listrik selesai praktikum sebelum meninggalkan tempat praktikum. Faktor kerapian akan mendapatkan penilaian khusus.

## PETUNJUK KHUSUS KESELAMATAN DI LABORATORIUM INFORMATIKA DAN KOMPUTER

### LINDUNGI DIRIMU

Laboratorium memiliki sekumpulan komputer dan juga perangkat keras jaringan yang memiliki banyak kabel dan menggunakan listrik. Yakinkan setiap peserta praktikum melakukan atau memenuhi beberapa hal sebagai berikut.

1. Dalam keadaan sehat, tidak kurang tidur, dan menyalakan lampu sesuai dengan penerangan cahaya di laboratorium.
2. Hindari menatap komputer dalam jangka waktu lama, setidaknya setiap 60 menit arahkan mata anda keluar dari layar komputer (misalnya jendela, pemandangan di luar jendela).
3. Hindari mengetik dalam jangka waktu yang lama tanpa berhenti, setidaknya setiap 30 menit regangkan pergelangan tangan dan jari.
4. Tetap menjaga postur tubuh pada saat duduk di depan komputer dengan nyaman.
5. Tidak diizinkan membuka komputer, atau perangkat lain yang terhubung di dalamnya, termasuk kabel-kabel jaringan terkecuali atas petunjuk profesional dan tercatat pada modul praktikum.

### LINDUNGI PERANGKAT PRAKTIKUM

Laboratorium memiliki sekumpulan komputer dan juga perangkat keras jaringan yang memiliki banyak kabel dan menggunakan listrik. Yakinkan pada beberapa hal berikut ini pada saat mengoperasikannya.

1. Tidak membawa makanan dan minuman di dalam laboratorium.
2. Selalu mematikan komputer melalui sistem operasi dan tidak mencabut kabel power tanpa berdiskusi dengan laboran.
3. Tidak memindah-mindahkan kabel jaringan yang terpasang di setiap komputer.
4. Matikan jika komputer tidak digunakan.
5. Tidak memindahkan berkas melalui flashdisk tanpa izin laboran atau asisten. Setiap asisten berhak melakukan pemindaian berkas untuk menghindari ancaman keamanan, berkas-berkas hasil praktikum disarankan diarsipkan di email atau di cloud storage institusi seperti di <http://365.ugm.ac.id>.
6. Pastikan perangkat lunak praktikum dapat berfungsi dengan baik, hubungi segera asisten untuk memperbaiki jika terdapat kerusakan di perangkat lunak praktikum.

## MODUL 0

### A: SETTING ENVIRONMENT

#### 1: TUJUAN PRAKTIKUM

Tujuan pada praktikum kali ini adalah :

1. Persiapan sebelum dapat menggunakan *compiler* (yang akan dijelaskan pada Unit 1)
2. Persiapan untuk praktikum pertama

#### 2: MEMPERSIAPKAN PERANGKAT LUNAK

Pada lab ini silahkan memasang perangkat lunak yang dibutuhkan untuk kebutuhan pembelajaran

1. PC/Laptop
2. OS: GNU/Linux, Windows 7 atau lebih baru, macOS 10.14 atau lebih baru
3. Administrator/Root Access

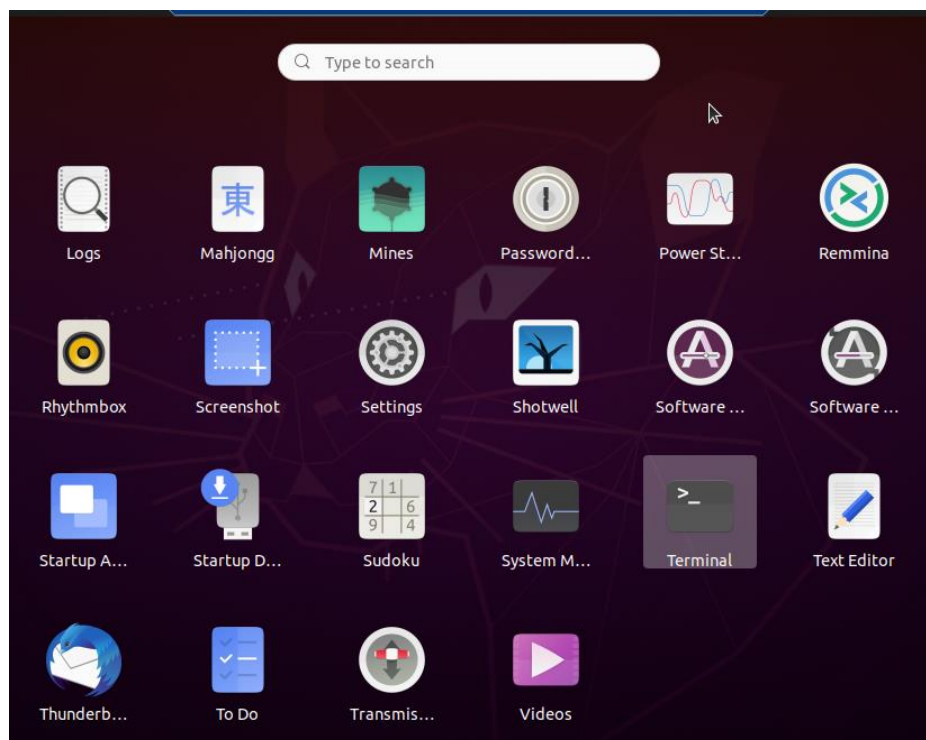
#### 3: LANGKAH PERCOBAAN

##### PADA OS LINUX

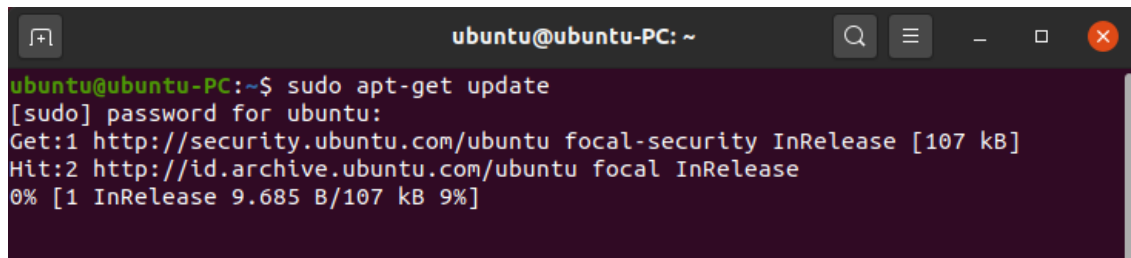
##### Mengaktifkan Windows Subsystem for Linux menggunakan PowerShell

Tutorial akan menggunakan Ubuntu 20.04 ini juga dapat diaplikasikan ke distro basis Debian lainnya. Langkah-langkah yang perlu diikuti adalah :

1. Buka **Terminal Emulator**



2. Perbarui *database repository* lokal dengan mengetik perintah berikut, kemudian tekan **ENTER**  
| **\$ sudo apt-get update**

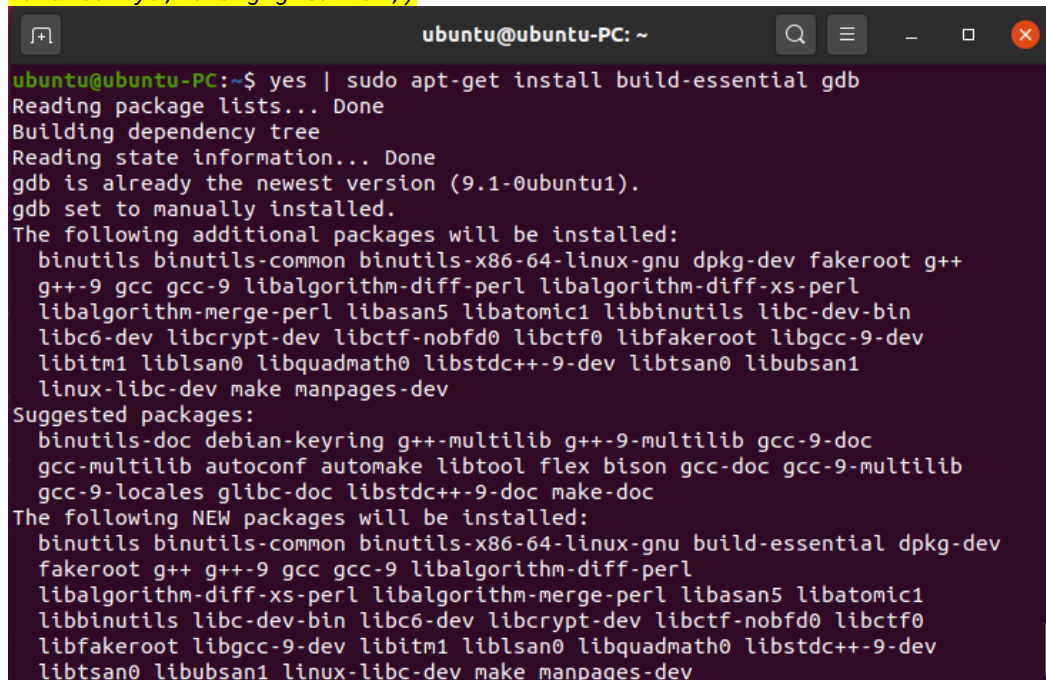


```
ubuntu@ubuntu-PC: ~  
ubuntu@ubuntu-PC:~$ sudo apt-get update  
[sudo] password for ubuntu:  
Get:1 http://security.ubuntu.com/ubuntu focal-security InRelease [107 kB]  
Hit:2 http://id.archive.ubuntu.com/ubuntu focal InRelease  
0% [1 InRelease 9.685 B/107 kB 9%]
```

Notes : Ingat, dollar sign (\$) TIDAK perlu diketik di terminal.

3. Ketikkan perintah berikut, kemudian tekan **ENTER**  
| **\$ sudo apt-get install build-essential gdb**

Notes : Masukkan password, jangan risau kalau sewaktu ngetik ga keluar karakternya, emang gitu kok;)



```
ubuntu@ubuntu-PC: ~  
ubuntu@ubuntu-PC:~$ yes | sudo apt-get install build-essential gdb  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
gdb is already the newest version (9.1-0ubuntu1).  
gdb set to manually installed.  
The following additional packages will be installed:  
binutils binutils-common binutils-x86-64-linux-gnu dpkg-dev fakeroot g++  
g++-9 gcc gcc-9 libalgorithm-diff-perl libalgorithm-diff-xs-perl  
libalgorithm-merge-perl libasan5 libatomic1 libbinutils libc-dev-bin  
libc6-dev libcrypt-dev libctf-nobfd0 libctf0 libfakeroot libgcc-9-dev  
libitm1 liblsan0 libquadmath0 libstdc++-9-dev libtsan0 libubsan1  
linux-libc-dev make manpages-dev  
Suggested packages:  
binutils-doc debian-keyring g++-multilib g++-9-multilib gcc-9-doc  
gcc-multilib autoconf automake libtool flex bison gcc-doc gcc-9-multilib  
gcc-9-locales glibc-doc libstdc++-9-doc make-doc  
The following NEW packages will be installed:  
binutils binutils-common binutils-x86-64-linux-gnu build-essential dpkg-dev  
fakeroot g++ g++-9 gcc gcc-9 libalgorithm-diff-perl  
libalgorithm-diff-xs-perl libalgorithm-merge-perl libasan5 libatomic1  
libbinutils libc-dev-bin libc6-dev libcrypt-dev libctf-nobfd0 libctf0  
libfakeroot libgcc-9-dev libitm1 liblsan0 libquadmath0 libstdc++-9-dev  
libtsan0 libubsan1 linux-libc-dev make manpages-dev
```

4. Ketik perintah berikut untuk mengkonfirmasi instalasi:  
| **\$ g++ -v**  
  
| **\$ gdb -v**

Notes : Output perintah akan sangat bervariasi, pastikan saja outputnya bukan: Command <g++ atau gdb> not found. Kalau muncul output seperti itu, ulangi lagi prosesnya.

## PADA OS WINDOWS

Untuk menginstall GCC dan GDB di OS Windows 64-bit gunakan langkah berikut:

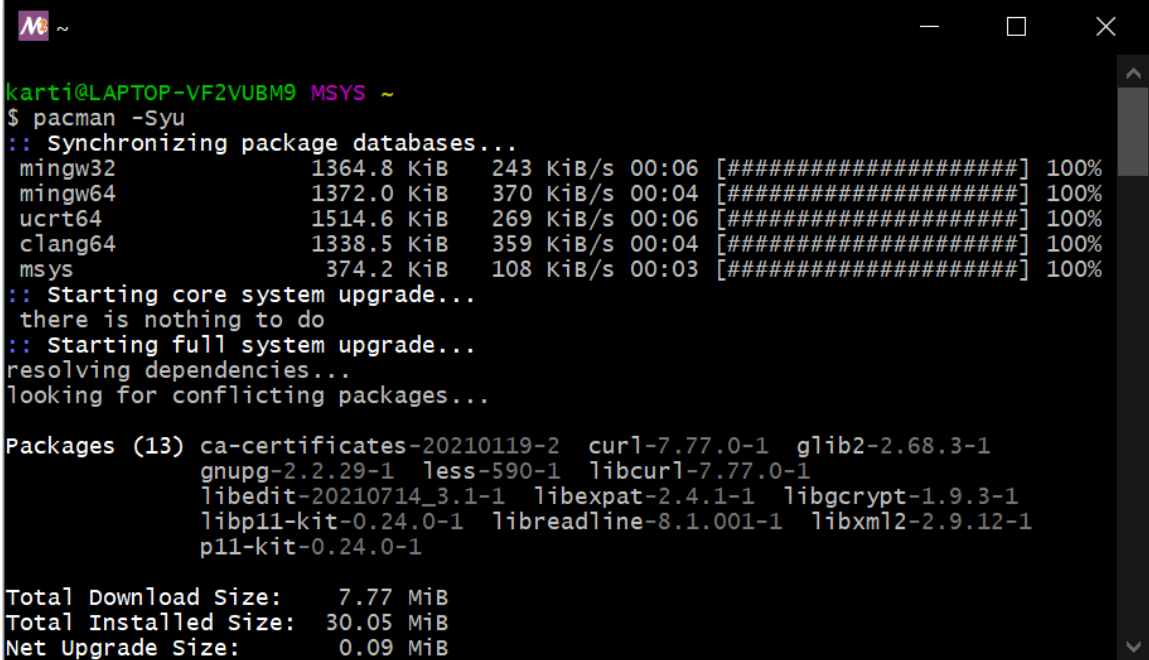
1. Pastikan tipe arsitektur OS merupakan 64 Bit dengan cara masuk ke **Control Panel > System and Security > System**.

System type: 64-bit Operating System, x64-based processor

2. Download **MSYS2** terbaru dari melalui pranala  
[http://repo.msys2.org/distrib/x86\\_64/](http://repo.msys2.org/distrib/x86_64/)  
Ambil versi terbaru yaitu [msys2-x86\\_64-20210725.exe](#)
3. Jalankan proses instalasi sampai selesai, biarkan opsi-opsi secara default.
4. Jalankan **MSYS2 MSYS** dari Start Menu.
5. Perbarui basis data repositori dan semua *binary* yang ada di sistem dengan mengetikkan perintah

| \$ **pacman -Syu**

Jika diminta konfirmasi, ketik **Y** lalu <Enter>.



```

karti@LAPTOP-VF2VUBM9 MSYS ~
$ pacman -Syu
:: Synchronizing package databases...
mingw32             1364.8 KiB   243 KiB/s   00:06 [#####] 100%
mingw64             1372.0 KiB   370 KiB/s   00:04 [#####] 100%
ucrt64              1514.6 KiB   269 KiB/s   00:06 [#####] 100%
clang64             1338.5 KiB   359 KiB/s   00:04 [#####] 100%
msys                 374.2 KiB   108 KiB/s   00:03 [#####] 100%
:: Starting core system upgrade...
there is nothing to do
:: Starting full system upgrade...
resolving dependencies...
looking for conflicting packages...

Packages (13) ca-certificates-20210119-2  curl-7.77.0-1  glib2-2.68.3-1
              gnupg-2.2.29-1  less-590-1  libcurl-7.77.0-1
              libedit-20210714_3.1-1  libexpat-2.4.1-1  libgcrypt-1.9.3-1
              libp11-kit-0.24.0-1  libreadline-8.1.001-1  libxml2-2.9.12-1
              p11-kit-0.24.0-1

Total Download Size:   7.77 MiB
Total Installed Size: 30.05 MiB
Net Upgrade Size:      0.09 MiB
  
```

6. Untuk menginstall GCC dan GDB, jalankan perintah:  
| \$ **pacman -S gcc gdb**
7. Periksa instalasi dengan memasukkan perintah  
| \$ **gcc --version**  
| \$ **gdb --version**

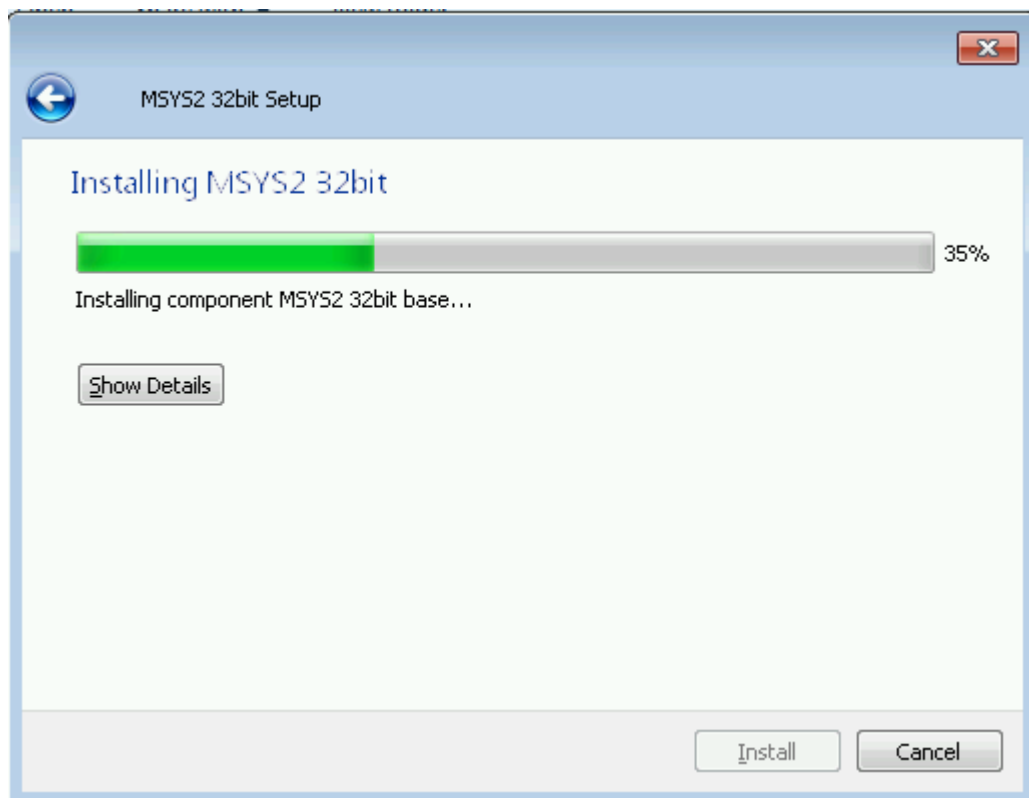
Untuk menginstall GCC dan GDB di OS Windows 32-bit gunakan langkah berikut:

1. Pastikan tipe arsitektur OS merupakan 32-bit. Pergi ke **Control Panel > System and Security > System**.

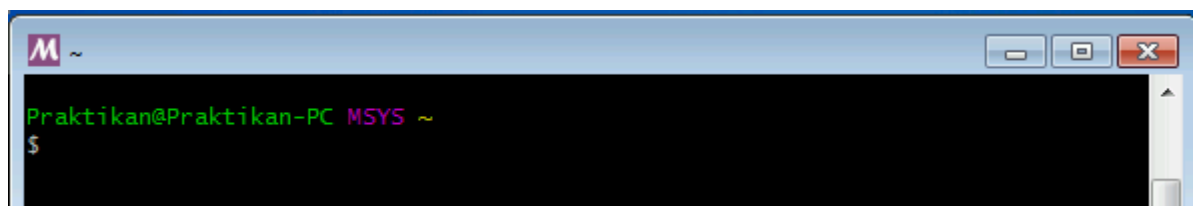
System type: 32-bit Operating System

2. Download **MSYS2** terbaru melalui pranala  
<http://repo.msys2.org/distrib/i686/>  
Ambil versi terbaru yaitu [msys2-i686-20200517.exe](#)
3. Jalankan installer tersebut, biarkan parameter instalasi *default*.





4. Jalankan **MSYS2** sekarang.



5. [Khusus untuk 32-bit OS dengan MSYS2 20200517] *Import* keyring terbaru dengan menjalankan perintah
- ```
| $ curl -O http://repo.msys2.org/msys/x86_64/msys2-keyring-r21.b39fb11-1-any.pkg.tar.xz  
| $ curl -O http://repo.msys2.org/msys/x86_64/msys2-keyring-r21.b39fb11-1-any.pkg.tar.xz.sig  
| $ pacman -U --config <(echo) msys2-keyring-r21.b39fb11-1-any.pkg.tar.xz
```

```

Praktikan@Praktikan-PC MSYS ~
$ pacman -U --config <(echo) msys2-keyring-r21.b39fb11-1-any.pkg.tar.xz
loading packages...
resolving dependencies...
looking for conflicting packages...

Packages (1) msys2-keyring-r21.b39fb11-1

Total Installed Size: 0.05 MiB
Net Upgrade Size: 0.03 MiB

:: Proceed with installation? [Y/n] Y
(1/1) checking keys in keyring [#####] 100%
(1/1) checking package integrity [#####] 100%
(1/1) loading package files [#####] 100%
(1/1) checking for file conflicts [#####] 100%
:: Processing package changes...
(1/1) upgrading msys2-keyring [#####] 100%
==> Appending keys from msys2.gpg...
gpg: Warning: using insecure memory!
==> Locally signing trusted keys in keyring...
-> Locally signing key 6E8FEAFF9644F54EED90EEA0790AE56A1D3CFDDC...
-> Locally signing key D55E7A6D7CE9BA1587C0ACACF40D263ECA25678A...
-> Locally signing key 123D4D51A1793859C2BE9168BBE514E53E0D0813...

```

6. Perbarui semua *binary* yang ada di sistem dengan melakukan perintah  
| **\$ pacman -Syu**
7. Install **GCC** dan **GDB** melalui perintah  
| **\$ pacman -S gcc gdb**  
Jika diminta untuk konfirmasi, masukkan **Y** lalu <Enter>  
Bila perlu, lakukan perintah baris pertama secara beberapa kali.

```

Praktikan@Praktikan-PC MSYS ~
$ pacman -S gcc gdb
resolving dependencies...
looking for conflicting packages...

Packages (12) binutils-2.34-2 expat-2.2.9-1 isl-0.22.1-1 mpc-1.1.0-1
mpdecimal-2.4.2-2 msys2-runtime-devel-3.0.7-6
msys2-w32api-headers-8.0.0.5683.629fd2b1-1
msys2-w32api-runtime-8.0.0.5683.629fd2b1-1 python-3.8.2-1
windows-default-manifest-6.4-1 gcc-9.3.0-1 gdb-9.1-1

Total Download Size: 55.14 MiB
Total Installed Size: 325.41 MiB

:: Proceed with installation? [Y/n] y
:: Retrieving packages...
binutils-2.34-2-i686 4.5 MiB 218 KiB/s 00:21 [#####] 100%
isl-0.22.1-1-i686 512.9 KiB 114 KiB/s 00:05 [#####] 100%
mpc-1.1.0-1-i686 68.7 KiB 79.9 KiB/s 00:01 [#####] 100%
msys2-runtime-de... 5.1 MiB 251 KiB/s 00:21 [#####] 100%
msys2-w32api-hea... 4.5 MiB 251 KiB/s 00:18 [#####] 100%
msys2-w32api-run... 1176.0 KiB 229 KiB/s 00:05 [#####] 100%
windows-default-... 1392.0 B 0.00 B/s 00:00 [#####] 100%
gcc-9.3.0-1-i686 1988.7 KiB 178 KiB/s 02:18 [#-----] 7%

```

8. Konfirmasi bahwa instalasi telah sukses dengan memasukkan perintah  
| **\$ gcc -v**  
| **\$ gdb -v**

```

Praktikan@Praktikan-PC MSYS ~
$ gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/lib/gcc/i686-pc-msys/9.3.0/lto-wrapper.exe
Target: i686-pc-msys
Configured with: /msysdev/gcc/src/gcc-9.3.0/configure --build=i686-pc-msys --pre
fix=/usr --libexecdir=/usr/lib --enable-bootstrap --enable-shared --enable-share
d-libgcc --enable-static --enable-version-specific-runtime-libs --with-arch=i686
--with-tune=generic --disable-multilib --disable-sjlj-exceptions --enable-__cxa
_atexit --with-dwarf2 --enable-languages=c,c++,fortran,lto --enable-graphite --e
nable-threads=posix --enable-libatomic --enable-libgomp --disable-libitm --enabl
e-libquadmath --enable-libquadmath-support --disable-libssp --disable-win32-regi
stry --disable-symvers --with-gnu-ld --with-gnu-as --disable-lsl-version-check --
enable-checking=release --without-libiconv-prefix --without-libintl-prefix --wi
th-system-zlib --enable-linker-build-id --with-default-libstdcxx-abi=gcc4-compat
ible --enable-libstdcxx-filesystem-ts
Thread model: posix
gcc version 9.3.0 (GCC)

Praktikan@Praktikan-PC MSYS ~
$ gdb -v
GNU gdb (GDB) 9.1
Copyright (C) 2020 Free Software Foundation, Inc.

```

## PADA OS : macOS

Untuk menginstall GNU C Compiler dan GNU Debugger pada macOS, gunakan langkah-langkah berikut :

1. Buka **Terminal App** dengan cara masuk ke **Launchpad > Other > Terminal** atau bisa melalui **Spotlight**, ketik **Terminal**.

```

praktikan@Praktikans-MBP ~$

```

2. Ketik perintah berikut untuk menginstal **Xcode Command Line Tools**. C Compiler juga akan terinstall melalui perintah ini.

| \$ **xcode-select --install**

```

praktikan@Praktikans-MBP ~$ xcode-select --install

```

**Notes : Ingat, dollar sign (\$) TIDAK perlu diketik di terminal. Masukkan password kalau diminta.**

3. Install Homebrew Package Manager dengan cara buka laman <https://brew.sh> kemudian salin perintah instalasi tersebut pada Terminal App

```

praktikan@Praktikans-MBP ~$ /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install.sh)"
Password:

```

```
$ /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install.sh)"
```

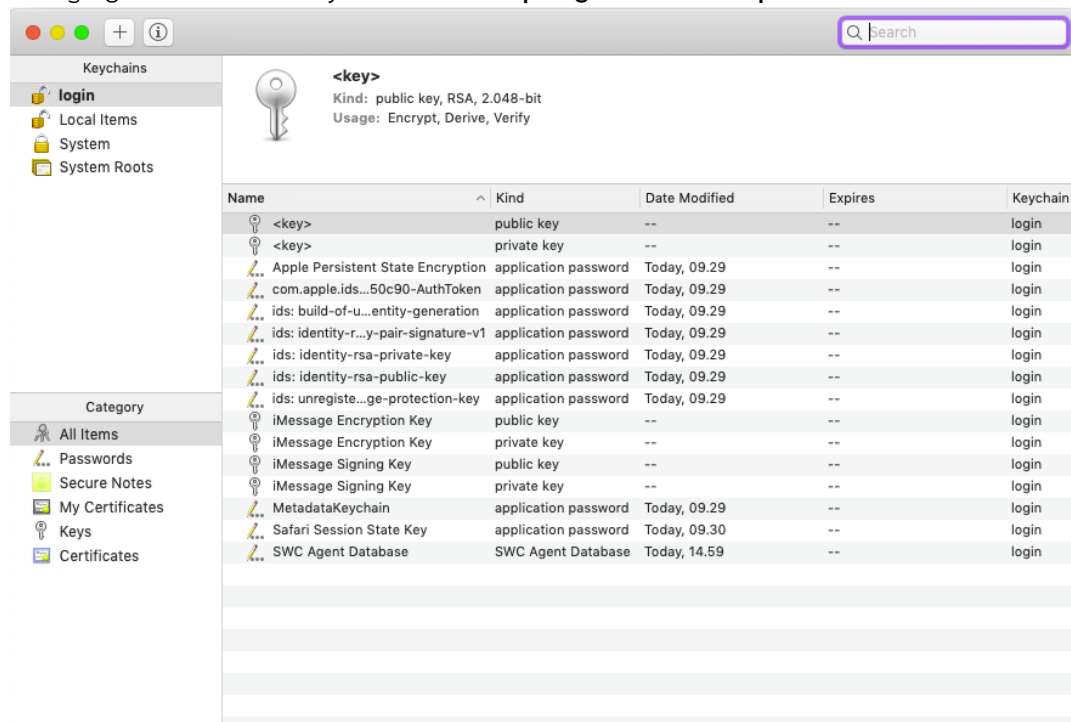
4. Install gcc dan gdb dengan Homebrew melalui perintah  
| **\$ brew install gcc**

*Grab your coffee, it's going to be a long run;)*

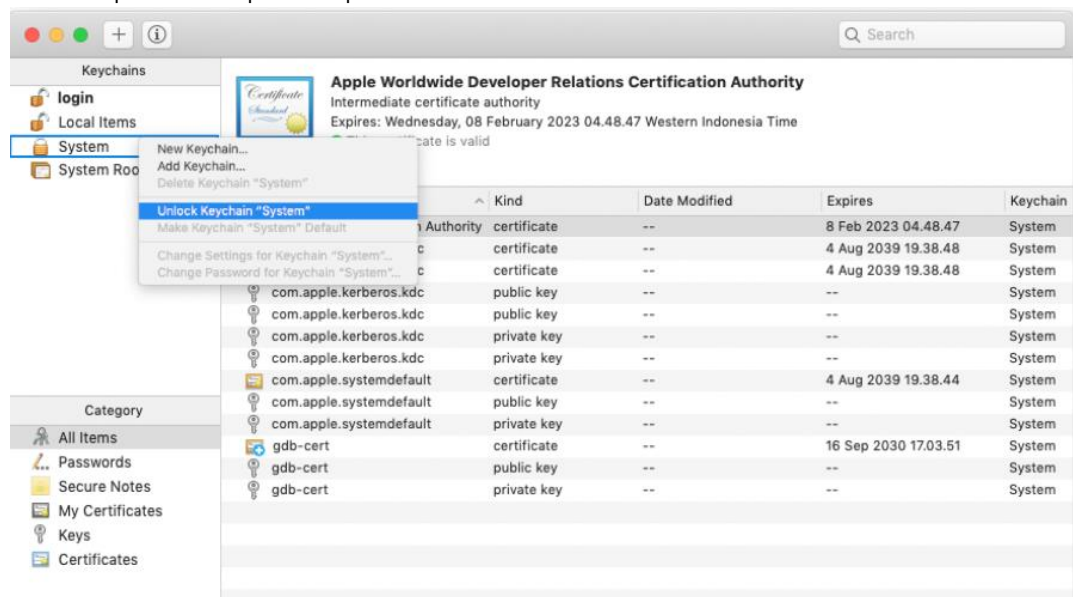
```
[praktikan@Praktikans-MBP Documents$ brew install gcc gdb]
Updating Homebrew...
==> Downloading https://homebrew.bintray.com/bottles/gmp-6.2.0.mojave.bottle.tar
==> Downloading from https://d29vzk4ow07wi7.cloudfront.net/1bba4983a4c883c8eb8b
##### 100.0%
==> Downloading https://homebrew.bintray.com/bottles/isl-0.22.1.mojave.bottle.ta
==> Downloading from https://d29vzk4ow07wi7.cloudfront.net/29213891860c971e084d1
##### 100.0%
==> Downloading https://homebrew.bintray.com/bottles/mpfr-4.1.0.mojave.bottle.ta
==> Downloading from https://d29vzk4ow07wi7.cloudfront.net/93c0d2ca093819f125300
##### 100.0%
==> Downloading https://homebrew.bintray.com/bottles/libmpc-1.2.0.mojave.bottle.
##### 100.0%
==> Downloading https://ftp.gnu.org/gnu/gcc/gcc-10.2.0/gcc-10.2.0.tar.xz
##### 100.0%
Warning: Your Xcode (11.2.1) is outdated.
Please update to Xcode 11.3.1 (or delete it).
Xcode can be updated from the App Store.

==> Installing dependencies for gcc: gmp, isl, mpfr and libmpc
==> Installing gcc dependency: gmp
==> Pouring gmp-6.2.0.mojave.bottle.tar.gz
📦 /Users/praktikan/homebrew/Cellar/gmp/6.2.0: 20 files, 3.2MB
==> Installing gcc dependency: isl
```

5. Codesign gdb, buka Akses Keychain (Melalui Spotlight atau Launchpad)

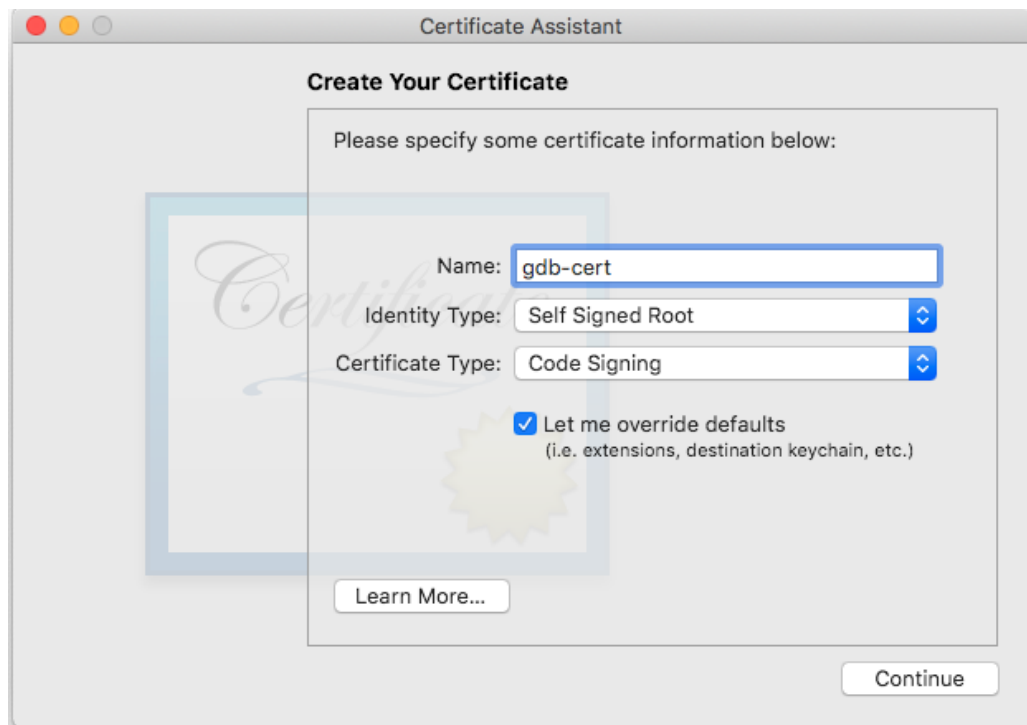


6. *Unlock* Keychain System dengan menekan **Ctrl+Click** pada System di sebelah Sidebar, kemudian masukkan password apabila diperlukan

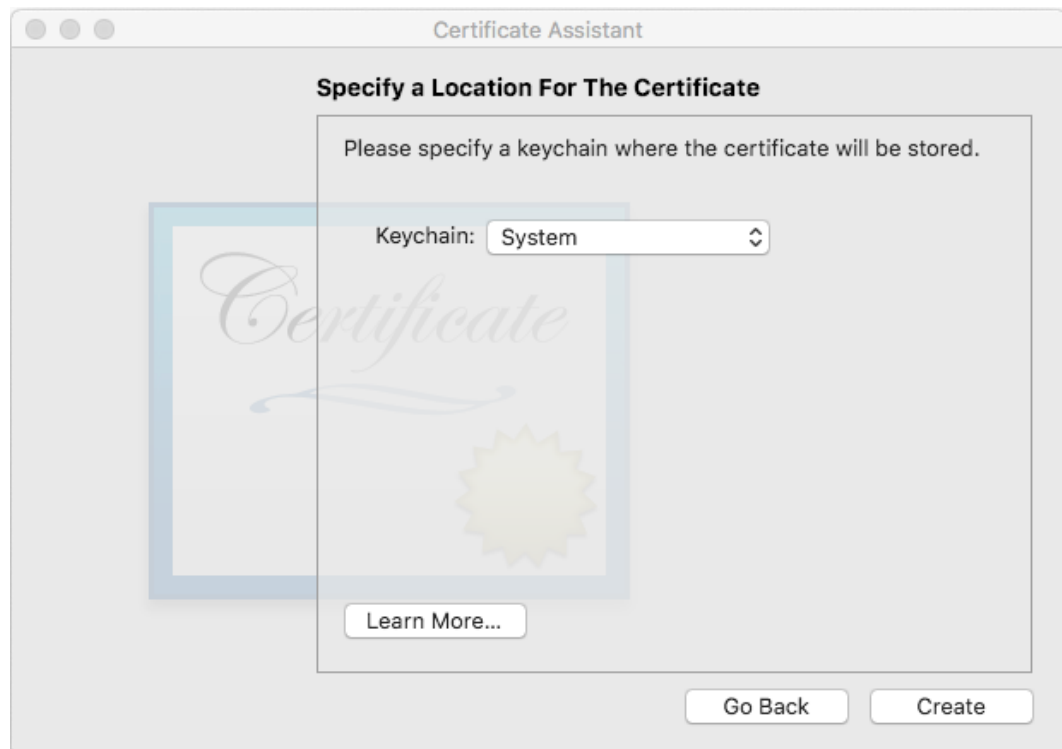


7. Melalui **Menubar**, buka **Keychain Access > Certificate Assistant > Create a Certificate**. Kemudian atur dengan format seperti di bawah ini :

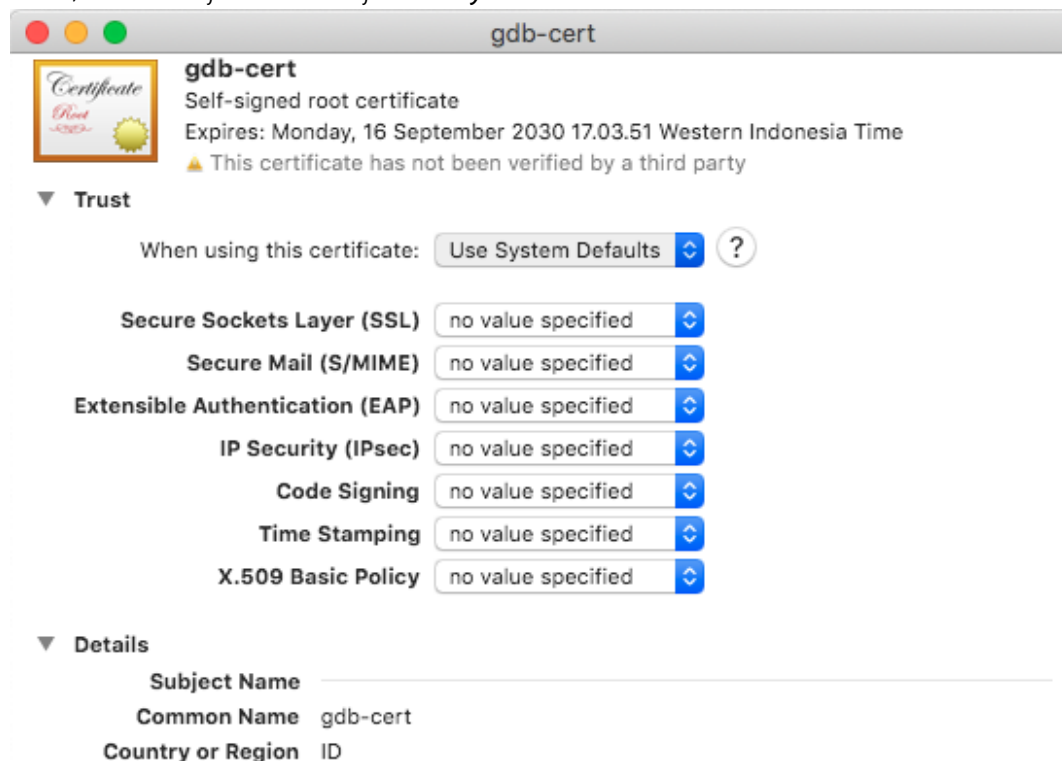
Name : gdb-cert  
 Identity Type : Self-Signed Root,  
 Cert Type : Code Signing,  
 Let me override defaults : check.



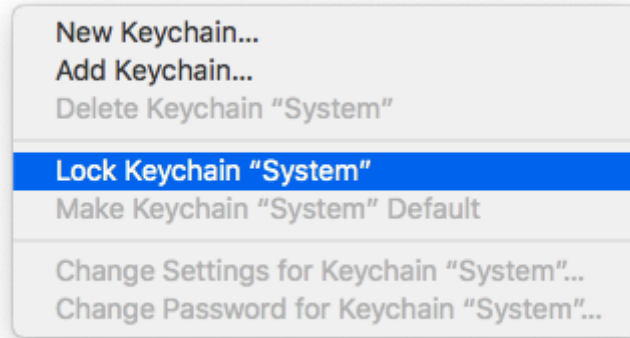
8. Lanjutkan instalasi (isi sesuai kebutuhanmu) sampai keluar perintah berupa **"Please specify a keychain ..."** kemudian pilih **System**. Setelah itu tekan **Create**.



9. Klik kanan pada Certificate yang baru saja terbuat, kemudian tekan **Get Info**. Pada bagian **Trust Section**, set semua jawaban menjadi **Always Trust**.



10. Klik kanan pada System Keychain pada sidebar, kemudian pilih **Lock Keychain "System"**. Dan kemudian segera lakukan reboot pada mac Anda.



11. Buat file baru bernama gdb.entitlements.xml dengan mengetikkan perintah seperti dibawah ini  
 | `$ curl -s http://ix.io/2xYe > gdb-entitlements.xml`

```

gdb-entitlements.xml
gdb-entitlements.xml > No Selection
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
3 "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
4 <plist version="1.0">
5 <dict>
6   <key>com.apple.security.cs.allow-jit</key>
7   <true/>
8
9   <key>com.apple.security.cs
10  .allow-unsigned-executable-memory</key>
11  <true/>
12
13  <key>com.apple.security.cs
14  .allow-dyld-environment-variables</key>
15  <true/>
16
17  <key>com.apple.security.cs
18  .disable-library-validation</key>
19  <true/>
20
21  <key>com.apple.security.cs
22  .disable-executable-page-protection</key>
23  <true/>
24
25  <key>com.apple.security.cs.debugger</key>
26  <true/>
27
28  <key>com.apple.security.get-task-allow</key>
29  <true/>
30 </dict>
31 </plist>
    
```

12. Ketik perintah berikut ini untuk masuk pada gdb  
 | `$ sudo codesign --entitlements gdb-entitlements.xml -fs gdb-cert $(which gdb)`



```
praktikan — sudo — 80x24
praktikan@Praktikans-MBP ~$ sudo codesign --entitlements gdb-entitlements.xml -f]
s gdb-cert $(which gdb)
Password: [?]
```

13. Ketik perintah berikut ini, kemudian restart Terminal anda  
 | \$ echo 'set startup-with-shell off' > ~/.gdbinit

```
praktikan — -bash — 80x24
praktikan@Praktikans-MBP ~$ echo 'set startup-with-shell off' > ~/.gdbinit
praktikan@Praktikans-MBP ~$ echo 'alias gcc="'$(which gcc-10)' '-ggdb"' >> ~/.bas]
h_profile
praktikan@Praktikans-MBP ~$ █
```

14. Jalankan perintah berikut ini untuk memeriksa apakah instalasi berhasil

```
| $ gcc -v
| $ gdb -v
```

```
praktikan — -bash — 80x27
Last login: Fri Sep 18 17:09:33 on ttys000
praktikan@Praktikans-MBP ~$ gcc -v
Using built-in specs.
COLLECT_GCC=/Users/praktikan/homebrew/bin/gcc-10
COLLECT_LTO_WRAPPER=/Users/praktikan/homebrew/Cellar/gcc/10.2.0/libexec/gcc/x86_
64-apple-darwin18/10.2.0/lto-wrapper
Target: x86_64-apple-darwin18
Configured with: ../configure --build=x86_64-apple-darwin18 --prefix=/Users/prak
tikan/homebrew/Cellar/gcc/10.2.0 --libdir=/Users/praktikan/homebrew/Cellar/gcc/1
0.2.0/lib/gcc/10 --disable-nls --enable-checking=release --enable-languages=c,c+
+,objc,obj-c++,fortran --program-suffix=-10 --with-gmp=/Users/praktikan/homebrew
/opt/gmp --with-mpfr=/Users/praktikan/homebrew/opt/mpfr --with-mpc=/Users/prakti
kan/homebrew/opt/libmpc --with-isl=/Users/praktikan/homebrew/opt/isl --with-syst
em-zlib --with-pkgversion='Homebrew GCC 10.2.0' --with-bugurl=https://github.com
/Homebrew/homebrew-core/issues --disable-multilib --with-native-system-header-di
r=/usr/include --with-sysroot=/Library/Developer/CommandLineTools/SDKs/MacOSX10.
14.sdk SED=/usr/bin/sed
Thread model: posix
Supported LTO compression algorithms: zlib
gcc version 10.2.0 (Homebrew GCC 10.2.0)
praktikan@Praktikans-MBP ~$ gdb -v
GNU gdb (GDB) 9.2
Copyright (C) 2020 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
praktikan@Praktikans-MBP ~$ █
```

## Troubleshooting : macOS

Untuk menginstall GNU C Compiler dan GNU Debugger pada macOS, gunakan langkah-langkah berikut :

1. (No debugging symbols found in a.out)  
Refer to this slide. Otherwise, always use -ggdb command flags when compiling.
2. please check gdb is codesigned - see taskgated(8)  
Check the code signing process.
3. macOS High Sierra has slightly different process on codesigning gdb. Read further.
4. Don't hesitate to ask the Lab. Assistant.



\*Versi baru ubuntu (diatas 18.04) biasanya menjalankan Software Updater sebagai background proses.

### **Text Editor**

Text editor yang digunakan pada praktikum ini adalah bebas, dengan kata lain praktikan diperbolehkan menggunakan text editor apapun apabila sudah terbiasa dengan tool tertentu.

Untuk yang belum terbiasa melakukan kegiatan programming, boleh menggunakan beberapa text editor ringan berikut:

- **Notepad++ untuk Windows**

Dapat diunduh melalui [Notepad++](#)

- **Gedit untuk Linux**

Dapat diinstal dengan cara

```
sudo apt install gedit
```

- **Textmate untuk MacOS**

Dapat diunduh melalui <https://github.com/textmate/textmate>

Atau jika malas menginstall aplikasi tambahan, sebenarnya menggunakan Text Editor bawaan setiap OS sudah cukup, hanya saja tidak ada bantuan seperti syntax highlighting atau sejenisnya

### **4: REFERENSI**

1. <https://gcc.gnu.org/onlinedocs/gcc/>
2. <https://sourceware.org/gdb/wiki/PermissionsDarwin>

## B : PENGENALAN GCC

### 1: TUJUAN PRAKTIKUM

Tujuan pada praktikum kali ini adalah :

1. Memahami fungsi dan kegunaan dasar gcc.
2. Mampu menggunakan gcc untuk membuat program dalam bahasa C++.

### 2: DASAR TEORI

Pada GNU Compiler Collection (GCC) adalah sebuah program yang berupa kumpulan compiler berbagai bahasa pemrograman seperti C, C++, Fortran, dan lain lain. Implementasi GCC yang paling luas saat ini adalah untuk meng-compile program program yang ditulis untuk bahasa C dan bahasa C++.

#### HINT

Kita dapat mengubah source code dari GCC, seperti menambah patch, dsb. Dengan demikian dimungkinkan untuk menjalankan program yang di-compile dengan GCC ini dalam sistem operasi Windows

Sebagai sebuah compiler bahasa pemrograman, GCC memiliki beberapa keuntungan, antara lain sebagai berikut:

1. Tersedia luas
2. Bersifat *Open Community*
3. Dapat digunakan sebagai *cross-compiler*

Untuk meng--compile sebuah *source code* yang ditulis dalam bahasa C++, digunakan perintah `g++`. Sebagai contoh, untuk meng-compile program yang ditulis dalam file `coba.cpp` digunakan perintah sebagai berikut:

```
$ g++ coba.cpp
```

Perintah di atas akan meng-compile `coba.cpp` menjadi suatu file executable bernama `a.out` (di linux) dan `a.exe` (di windows). Untuk mengganti nama file output standard `a.out` tersebut menjadi nama lain yang diinginkan, digunakan option `--o` dengan perintah sebagai berikut:

```
$ g++ --o [output] [source]
```

Selain langsung membuat executable file, GCC dapat juga digunakan untuk meng-compile source code dalam bahasa C++ menjadi sebuah object file berakhiran `.o` menggunakan option `--c`, dengan bentuk perintahnya adalah seperti berikut:

```
$ g++ --c [source]
```

Perintah di atas akan menghasilkan object file yang namanya sama dengan file source nya, tetapi akhirnya tidak lagi `.cpp` , melainkan `.o` .

## Linking

Linking adalah istilah yang sering digunakan untuk menyebut proses pembuatan suatu executable file dari satu atau lebih object file. Pada gcc, proses ini dilakukan dengan menggunakan option `--o`. Bentuk perintahnya adalah sebagai berikut:

```
$ g++ --o [executable file] [object-file 1] [object-file 2] ...
```

## Debugging

Selain untuk meng-compile source dan membuat executable, GCC juga memiliki fasilitas untuk melakukan debugging pada program yang dibuat, yaitu GDB (GNU DeBugger). Untuk menggunakan fasilitas debugging GDB, source code harus di-compile dengan menggunakan option `-g` untuk memasukkan informasi debugging ke dalam file executable yang dibuat. Bentuk perintahnya adalah sebagai berikut:

```
$ g++ -g --o [executable] [source]
```

Untuk melakukan debugging pada program yang telah di-compile dengan option `--g`, digunakan perintah `gdb` seperti di bawah ini:

```
$ gdb [executable]
```

Setelah mengetikkan perintah di atas, maka kita akan masuk ke command line `gdb`, di mana kita dapat memasukkan berbagai perintah debugging yang tersedia. Beberapa perintah debugging dasar ditunjukkan oleh tabel di bawah ini:

| Perintah          | Fungsi                           |
|-------------------|----------------------------------|
| <code>run</code>  | Menjalankan program              |
| <code>list</code> | Melihat di mana program berhenti |
| <code>quit</code> | Keluar dari <code>gdb</code>     |

### HINT

Untuk melihat fungsi-fungsi pada GCC lebih lanjut, dapat dilakukan dengan mengakses manual GCC, yaitu dengan mengetikkan `man g++`

## 3: LANGKAH PERCOBAAN

### Percobaan 1 : Compiling Program dengan GCC

1. Pada percobaan ini, tuliskan 2 program berikut menggunakan text editor dan simpan masing-masing dengan nama `myfirstprogram.cpp` dan `mysecondprogram.cpp` :

`myfirstprogram.cpp`

```
#include <iostream>

using namespace std;

int main(){
    cout << "this is my first program...\n";
}
```

`mysecondprogram.cpp`

```
#include <iostream>

using namespace std;

int main(){
    cout << "this is my second program...\n";
}
```

2. Setelah itu, bukalah console dan masuklah ke dalam direktori di mana kedua file tadi disimpan, kemudian masukkan perintah-perintah di bawah ini:

Linux OS:

```
$ g++ myfirstprogram.cpp
$ ./a.out
$ g++ mysecondprogram.cpp
$ ./a.out
```

Windows OS:

```
$ g++ myfirstprogram.cpp
$ ./a.exe
$ g++ mysecondprogram.cpp
$ ./a.exe
```

**BAGAIMANAKAH KELUARAN YANG DIHASILKAN OLEH MASING--MASING PERINTAH A.OUT (UNTUK LINUX) / A.EXE (UNTUK WINDOWS) DI ATAS? JELASKAN DALAM LAPORAN RESMI YANG ANDA BUAT!**

## **Percobaan 2 : Linking Object File**

1. Untuk percobaan ini, tuliskan kedua program berikut menggunakan editor dan simpanlah masing-masing dengan nama `linking1.cpp` dan `linking2.cpp` pada direktori Anda:

`linking1.cpp`

```
#include <iostream>

using namespace std;

void linking2();

int main(){
    cout << "first file to be linked... \n";
    linking2();
}
```

linking2.cpp

```
#include <iostream>

using namespace std;

void linking2(){
    cout << "second file to be linked...\n";
}
```

2. Setelah itu, jalankan perintah--perintah berikut ini di dalam direktori kerja Anda:

```
$ g++ -c linking1.cpp
$ g++ -c linking2.cpp
$ g++ -o linkedprogram linking1.o linking2.o
$ ./linkedprogram
```

**JELASKAN KEGUNAAN MASING--MASING PERINTAH DI ATAS DAN AMATI HASILNYA (TULISKAN DALAM LEMBAR LAPORAN SEMENTARA DI AKHIR BAB INI).**

3. Berikutnya, masukkan perintah--perintah berikut:

```
$ g++ --o linkedprogram linking1.o
$ ./linkedprogram
```

**BAGAIMANAKAH HASIL EKSEKUSI LINKEDPROGRAM KALI INI? JELASKAN ALASANNYA DALAM LAPORAN RESMI YANG ANDA BUAT.**

### **Percobaan 3 : Debugging**

1. Untuk percobaan kali ini, tulislah program berikut dan simpanlah pada direktori Anda dengan nama debug.cpp :

debug.cpp

```
#include <iostream>

using namespace std;

int main(){
    cout << "for debugging use only...\n";
}
```

2. Setelah itu, jalankan perintah--perintah berikut ini:

```
$ g++ -g -o debugfile debug.cpp
```

```
$ gdb debugfile
(gdb) run
```

Amati hasilnya.

3. Keluarlah dari command line gdb dengan perintah quit, kemudian masukkan perintah--perintah berikut:

```
$ g++ -o debugfile debug.cpp
$ gdb debugfile
(gdb) run
```

**ADAKAH PERBEDAAN HASIL EKSEKUSI DENGAN YANG SEBELUMNYA? MENGAPA?  
(JELASKAN DALAM LAPORAN RESMI ANDA!)**

#### 4: TUGAS

1. Program GCC adalah sebuah software program yang bersifat open source. Terangkan perbedaan antara software open source, freeware, dan shareware. Lampirkan jawaban / tugas ini pada laporan resmi yang akan Anda buat.
2. Jelaskan kegunaan perintah di bawah ini! Bagaimanakah hasilnya bila perintah tersebut di eksekusi? (lihat langkah percobaan 2).  

```
$ g++ -c linking1.cpp
$ g++ -c linking2.cpp
$ g++ -o linkedprogram linking1.o linking2.o
$ ./linkedprogram
```
3. Apakah perbedaan executable file dan object file?
4. Apakah yang anda ketahui tentang debugging dan apa gunanya?

#### 5: REFERENSI

Di bawah ini adalah referensi online dari UNIT I. Apabila mengalami kesulitan dalam memahami materi, silahkan buka referensi di bawah ini :

1. <http://www.gnu.org/software/gcc/>
2. <http://users.actcom.co.il/~choo/lupg/tutorials/c-on-unix/c-on-unix.html>
3. <http://web.cs.ucla.edu/classes/fall14/cs143/project/cpp/gcc-intro.html>
4. [https://gcc.gnu.org/onlinedocs/gcc/Invoking-G\\_002b\\_002b.html#Invoking-G\\_002b\\_002b](https://gcc.gnu.org/onlinedocs/gcc/Invoking-G_002b_002b.html#Invoking-G_002b_002b)

#### 6: SEKILAS MATERI UNIT BERIKUTNYA

Materi praktikum pemrograman dasar unit selanjutnya akan lebih banyak membahas beberapa hal, terkait dengan operasi dasar dalam pemrograman bahasa C++. Beberapa hal yang patut Anda ketahui, antara lain :

1. Struktur pemrograman bahasa C++
2. Tipe data dan operator
3. Operasi--operasi dasar dalam pemrograman dasar