

# Dasar-Dasar Pemrograman 2

## Getting Started with a Java Project



FAKULTAS  
ILMU  
KOMPUTER

### Adopted from:

- Getting Started with a Java Project, Tim Asisten Dosen DDP2 2019/2020 Genap
- Getting Started with a Java Project, Tim Asisten Dosen DDP2 2020/2021 Genap
- Getting Started with a Java Project, Tim Asisten Dosen DDP2 2021/2022 Genap

### What's new: Using GitHub as Git Repository ✨

---

Setelah berhasil membuat program Java sederhana pada lab, mari kita coba menjalankan sebuah proyek Java.

## Gradle

Terdapat beberapa masalah yang mungkin terjadi saat kita membangun suatu proyek Java yang besar:

1. Biasanya sebuah proyek membutuhkan **dependency** atau **library-library non-standar**. Mungkin saja sebuah proyek memerlukan ratusan *dependency* untuk diinstal oleh orang yang ingin terlibat dalam proyek tersebut. Tentunya hal ini sangat menyulitkan orang tersebut.
2. Setiap ada **perubahan** pada suatu *file*, maka seluruh bagian dari *file* yang berubah **harus dikompilasi**. Tidak mengkompilasi seluruh *file* memungkinkan program tidak berjalan sesuai ekspektasi. Namun, hal ini kurang optimal. Oleh karena itu, diperlukan *tools* yang bisa mengkompilasi *file* yang berubah saja.
3. Biasanya suatu proyek akan dilengkapi **script**, misalnya untuk menjalankan **test**. *Test* tersebut berguna untuk memastikan bahwa perubahan yang dilakukan *programmer* benar atau sesuai ekspektasi. Oleh karena itu, kita memerlukan *tools* untuk menjalankan *script* tersebut.

Masalah-masalah yang mungkin terjadi saat pembangunan proyek besar, mendukung diciptakannya sebuah *build tools* di mana salah satunya adalah **Gradle**.

## Git

Mengapa kita memerlukan Git? Misalnya kamu sedang membangun sebuah program. Kemudian kamu terpikirkan untuk menambah sebuah fitur baru. Namun, ternyata kode baru kamu tidak tepat dan mengakibatkan seluruh program tidak bisa dijalankan. Tentunya kamu ingin meng-*undo* perubahan yang telah kamu lakukan tersebut. Dengan adanya Git, kamu bisa mengembalikan kode tersebut ke versi sebelum adanya perubahan. **Git merupakan sebuah Version Control System**. Dengan Git, kamu dapat mengunggah kode kamu ke situs penyedia layanan *hosting* repositori Git, salah satunya GitHub.

# Proyek Java Pertamaku

## GitHub sebagai Git Repository

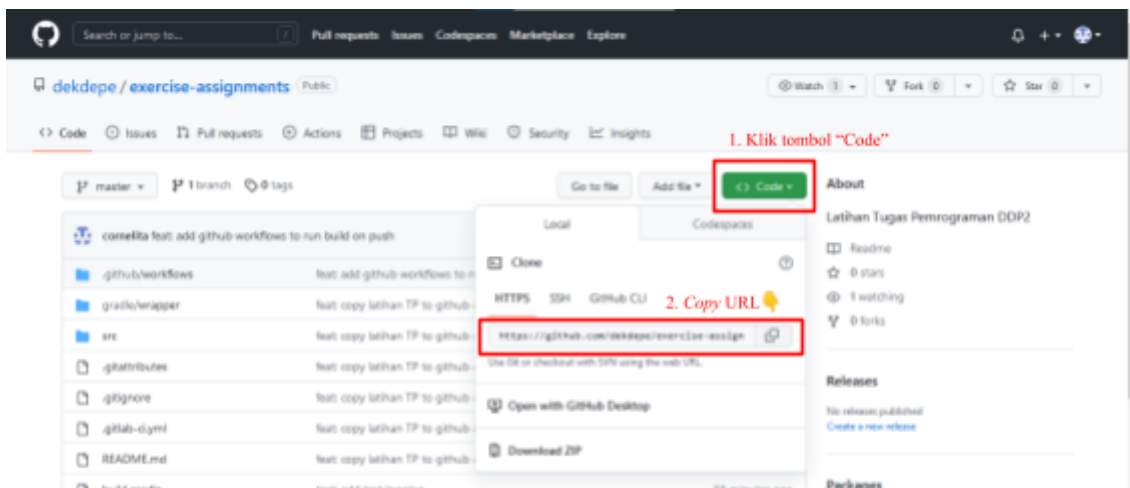
**Notes:** Panduan ini menggunakan Java versi 17 (JDK17)

Pada tutorial ini, kamu akan dipandu untuk menggunakan Git dengan cara mengunduh sebuah proyek Java, memodifikasinya, dan mengunggah perubahan tersebut ke GitHub. Sebelum memulai, **pastikan kamu telah memiliki akun [GitHub](#)**.

### A. Menduplikat GitHub Repository

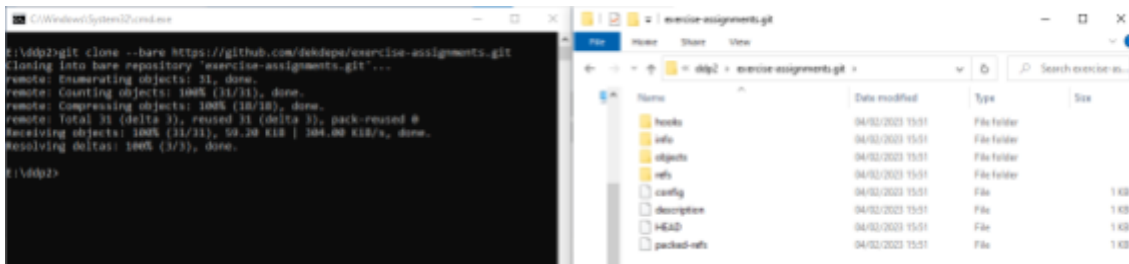
Pertama-tama kita akan menduplikat *repository* milik dekdepe ke akun GitHub milikmu.

1. Bukalah *repository* exercise-assignments milik akun GitHub dekdepe melalui [link ini](#). Kemudian *copy* URL-nya.



2. Bukalah *terminal/command prompt* (cmd) di perangkatmu, lalu masukkan perintah git clone dengan options bare untuk membuat [bare Git repository](#). Setelah melakukannya, kamu akan mendapatkan folder “exercise-assignments.git”.

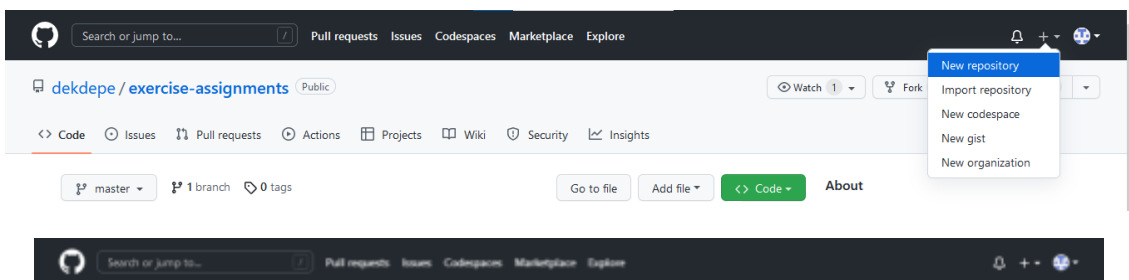




3. Buka folder “exercise-assignments.git” dengan menggunakan perintah `cd` (change directory).

```
E:\ddp2>cd exercise-assignments.git
E:\ddp2\exercise-assignments.git>
```

4. Buka GitHub lalu buatlah **private** repository baru yang akan menyimpan hasil duplikasi repository exercise-assignments milik dekdepe. Pastikan kamu memilih “Private” agar tidak semua orang dapat melihat proyekmu.



**Create a new repository**

A repository contains all project files, including the revision history. Already have a project repository elsewhere?  
[Import a repository.](#)

Repository template  
Start your repository with a template repository's contents.

No template

**Akun GitHubmu**

Owner \*

Repository name \*

Description (optional)

Public

Private

Initialize this repository with:

Skip this step if you're importing an existing repository.

Add a README file

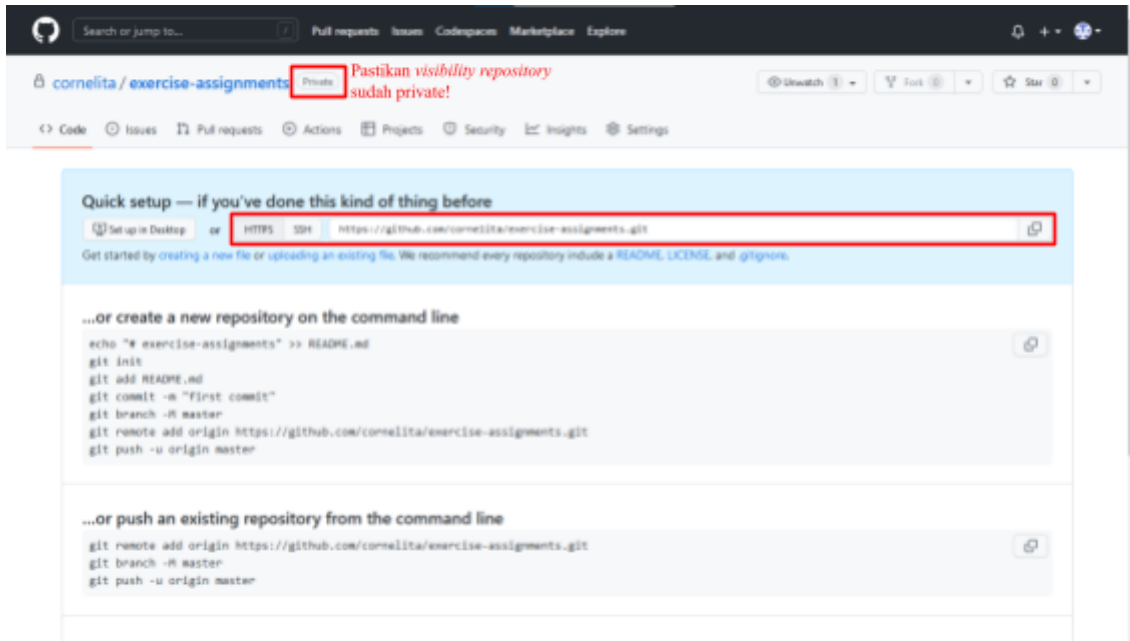
Add .gitignore

Choose which files not to track from a list of templates.

Choose a license

Create repository

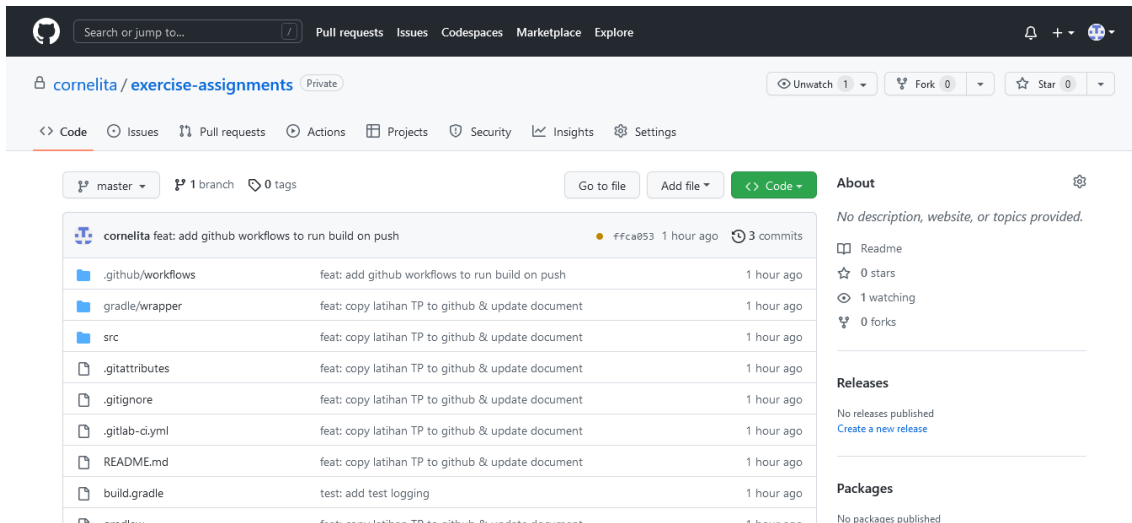
5. Kamu akan disambut dengan *repository* kosong seperti gambar di bawah ini. *Repository* ini akan menjadi *repository* pribadimu yang menyimpan hasil duplikat dari *repository* milik dekdepe. Copy URL yang disediakan.



6. Kembali ke *terminal*/cmd lalu lakukan push dengan options mirror ke *repository* yang baru kamu buat.



7. Kembali ke halaman GitHub *repository* yang kamu buat lalu lakukan *refresh*. Jika sudah berhasil melakukan duplikasi, maka akan muncul folder dan *file* yang sama seperti yang ada di *repository* milik dekdepe.

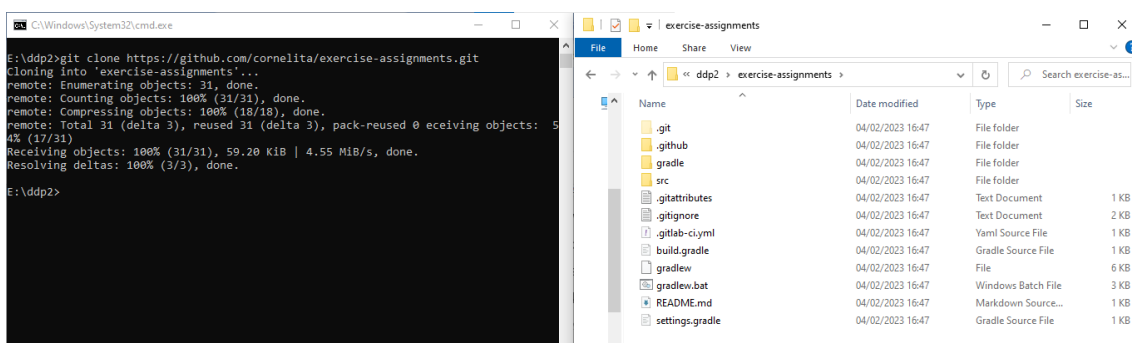


8. Setelah berhasil menduplikat *repository* *exercise-assignments* milik *dekdepe*, hapus folder “*exercise-assignments.git*” karena kita tidak memerlukannya lagi

## B. Clone Repository

Sekarang kamu telah mempunyai **private** *repository* baru yang merupakan hasil duplikasi *repository* milik *dekdepe*. Selanjutnya kita akan membuat *local copy* di perangkatmu dengan melakukan clone.

1. Buka *terminal/cmd* lalu lakukan *git clone* dengan menggunakan URL *repository*-mu seperti yang didapatkan pada bagian A langkah 5.



2. Buka proyek tersebut dengan text editor atau IDE favoritmu.

*Feel free to use, reuse, and share this work: the more we share, the more we have!*



### C. Menjalankan Java Program Menggunakan Gradle

Untuk menjalankan fungsi main yang berada pada file `src/main/java/exercise/Solver.java`, kamu dapat menggunakan *executable* Gradle Wrapper yaitu `gradlew.bat` yang berada pada proyek tersebut. Cukup jalankan perintah run menggunakan `gradlew.bat` pada terminal/cmd untuk menjalankan fungsi main tersebut.

**Note:** Jika kalian menggunakan macOS atau Linux, ganti seluruh perintah `gradlew.bat` pada dokumen ini menggunakan `./gradlew`, misalnya `./gradlew run`

```
E:\ddp2\exercise-assignments>gradlew.bat run

> Task :run
true
false

BUILD SUCCESSFUL in 2s
2 actionable tasks: 1 executed, 1 up-to-date
E:\ddp2\exercise-assignments>
```

### D. Menjalankan Test Menggunakan Gradle & Memperbaiki Program

Untuk menjalankan *test* yang berada pada `src/test/java/exercise/SolverTest.java`, kamu dapat menjalankan perintah test menggunakan `gradlew.bat` pada terminal/cmd. Test membantu *programmer* untuk memastikan bahwa kode yang dituliskan sesuai ekspektasi walaupun ada perubahan.

```
E:\ddp2\exercise-assignments>gradlew.bat test

> Task :test
SolverTest > testTwoSumNotConsiderSameNumber() FAILED
org.opentest4j.AssertionFailedError: Two sum should not consider the same number twice ==> expected: <false> but was: <true>
    at app//org.junit.jupiter.api.AssertionUtils.fail(AssertionUtils.java:55)
    at app//org.junit.jupiter.api.AssertFalse.assertFalse(AssertFalse.java:40)
    at app//org.junit.jupiter.api.Assertions.assertFalse(Assertions.java:235)
    at app//exercise.SolverTest.testTwoSumNotConsiderSameNumber(SolverTest.java:30)

5 tests completed, 1 failed

> Task :test FAILED

FAILURE: Build failed with an exception.

* What went wrong:
Execution failed for task ':test'.
> There were failing tests. See the report at: file:///E:/ddp2/exercise-assignments/build/reports/tests/test/index.html

* Try:
> Run with --stacktrace option to get the stack trace.
> Run with --info or --debug option to get more log output.
> Run with --scan to get full insights.

* Get more help at https://help.gradle.org

BUILD FAILED in 6s
3 actionable tasks: 2 executed, 1 up-to-date
```

Rupanya *test* kita gagal 🤔 Untuk mengetahui mengapa *test*-nya gagal, mari kita baca hasil dari menjalankan perintah test.

```
E:\ddp2\exercise-assignments>gradlew.bat test

> Task :test Penjelasn test yang gagal

SolverTest > testTwoSumNotConsiderSameNumber() FAILED
org.opentest4j.AssertionFailedError: Two sum should not consider the same number twice ==> expected: <false> but was: <true>
    at app//org.junit.jupiter.api.Assertions.fail(Assertions.java:55)
    at app//org.junit.jupiter.api.Assertions.assertFalse(Assertions.java:40)
    at app//org.junit.jupiter.api.Assertions.assertFalse(Assertions.java:235)
    at app//exercise.SolverTest.testTwoSumNotConsiderSameNumber(SolverTest.java:30)

5 tests completed, 1 failed ← Rangkuman hasil test

> Task :test FAILED

FAILURE: Build failed with an exception.

* What went wrong:
Execution failed for task ':test'.
> There were failing tests. See the report at: file:///E:/ddp2/exercise-assignments/build/reports/tests/test/index.html

* Try:
> Run with --stacktrace option to get the stack trace.
> Run with --info or --debug option to get more log output.
> Run with --scan to get full insights.

* Get more help at https://help.gradle.org

BUILD FAILED in 6s
3 actionable tasks: 2 executed, 1 up-to-date
```

Dari gambar di atas pada penjelasan test yang gagal dapat dilihat bahwa `testTwoSumNotConsiderSameNumber` **FAILED** dengan pesan berikut

```
org.opentest4j.AssertionFailedError: Two sum should not consider the same number
twice ==> expected: <false> but was: <true>
```

Pesan tersebut mengatakan bahwa “testTwoSumNotConsiderSameNumber” memiliki ekspektasi bahwa hasil *assert*-nya adalah false, tetapi yang didapatkan adalah true. Oleh karena itu *test*-nya gagal. Untuk mengetahui lebih lanjut mengenai *test*-nya, lihatlah kode *test* pada file `SolverTest.java`.

```
@Test
public void testTwoSumNotConsiderSameNumber() {
    assertFalse(new Solver().isExistTwoSum(new int[] {2}, 4), message: "Two sum should not consider the same number twice");
}
```

Angka 2 yang sama tidak boleh dipilih dua kali untuk menghasilkan 4. Wahh hal ini berarti kode pada fungsi `isExistTwoSum` belum tepat sehingga kamu perlu memperbaikinya. Coba perbaiki sesuai dengan pemahamanmu terlebih dahulu lalu jalankan perintah test lagi. Jika sudah berhasil diperbaiki maka ketika perintah test dijalankan, tampilannya akan menjadi seperti berikut.

PS: Jika kamu sudah menyerah, solusinya ada [disini](#).

```
E:\ddp2\exercise-assignments>gradlew.bat test

BUILD SUCCESSFUL in 3s
3 actionable tasks: 2 executed, 1 up-to-date
E:\ddp2\exercise-assignments>
```



## E. Menyimpan Perubahan ke GitHub

Kini kodemu telah sesuai ekspektasi karena sukses melalui seluruh test. Selanjutnya kita akan menyimpan perubahan tersebut ke GitHub dengan menggunakan git.

1. Pertama jalankan perintah `git status` untuk mengetahui *file* apa saja yang berubah.

```
E:\ddp2\exercise-assignments>git status
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   src/main/java/exercise/Solver.java

no changes added to commit (use "git add" and/or "git commit -a")
```

2. Untuk melihat perubahannya, jalankan perintah `git diff`. Untuk keluar dari `git diff`, tekan `q` (jika diperlukan).

```
E:\ddp2\exercise-assignments>git diff
diff --git a/src/main/java/exercise/Solver.java b/src/main/java/exercise/Solver.java
index 748b55b..ea65a7d 100644
--- a/src/main/java/exercise/Solver.java
+++ b/src/main/java/exercise/Solver.java
@@ -10,7 +10,7 @@ public class Solver {
    public boolean isExistTwoSum(int[] nums, int K) {
        for (int i = 0; i < nums.length; i++) {
            for (int j = 0; j < nums.length; j++) {
-                if (nums[i] + nums[j] == K) return true;
+                if (i != j && nums[i] + nums[j] == K) return true;
            }
        }
        return false;
    }
}
```

3. Tentukan file mana saja yang ingin disimpan perubahannya. Dalam kasus ini, hanya file `src/main/java/exercise/Solver.java` yang berubah. Kamu dapat menggunakan perintah `git add *`. Pastikan perubahan tersebut sudah ada di *staging* dan siap untuk di-*commit* dengan menjalankan perintah: `git status`.

```
E:\ddp2\exercise-assignments>git add *
E:\ddp2\exercise-assignments>git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   src/main/java/exercise/Solver.java
```

**Note:**

Ada beberapa karakter khusus yang dapat dimanfaatkan untuk mempersingkat perintah `git add`, misalnya:

- `git add *` : menambahkan semua *file* ke *staging*
- `git add .` : menambahkan semua *file* pada direktori saat ini beserta subdirektori ke *staging*
- `git add -u` : menambahkan hanya *file* yang berubah (bukan file baru) ke *staging*

4. Atur *username* dan *email* git di perangkatmu untuk menandai bahwa *commit* tersebut dibuat oleh kamu dengan menjalankan perintah berikut:

```
git config user.name "Nama Kamu"
```

```
git config user.email "emailkamu@server.tld"
```

Pastikan *email* yang digunakan sesuai dengan akun GitHub-mu.

**Note:** Tambahkan parameter `--global` jika kamu ingin mengatur konfigurasi secara global, tidak hanya untuk repositori tersebut.

```
git config --global user.name "Nama Kamu"
```

```
git config --global user.email "emailkamu@server.tld"
```

```
E:\ddp2\exercise-assignments>git config user.name "cornelita"
```


```
E:\ddp2\exercise-assignments>git config user.email "cornelitalugitasantoso@gmail.com"
```

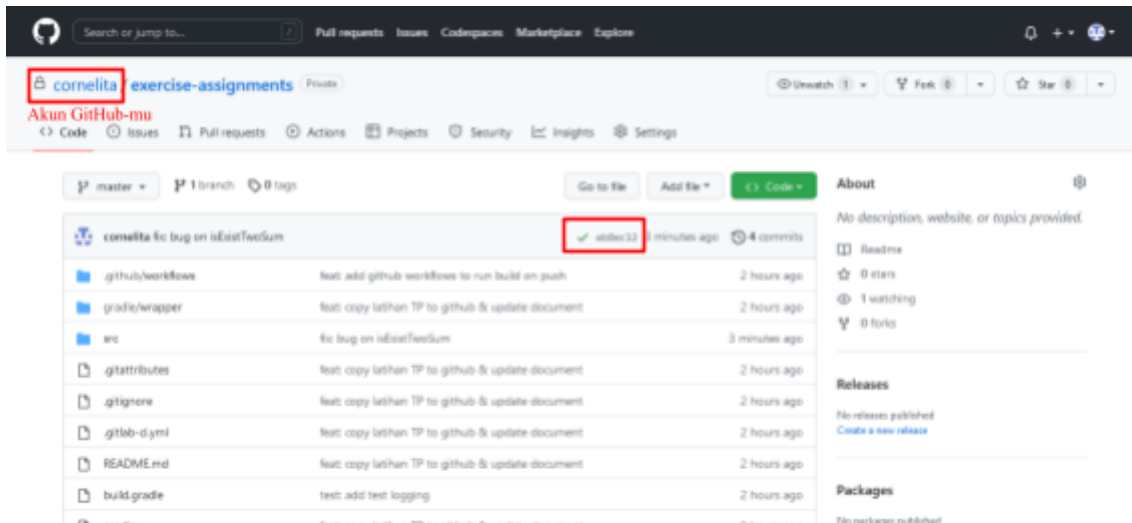
5. Simpan perubahan dengan menjalankan `git commit -m "<perubahan yang kamu lakukan>"`. Contoh:

```
E:\ddp2\exercise-assignments>git commit -m "fix: bug on isExistTwoSum"
[master eb8ec12] fix: bug on isExistTwoSum
1 file changed, 1 insertion(+), 1 deletion(-)
```

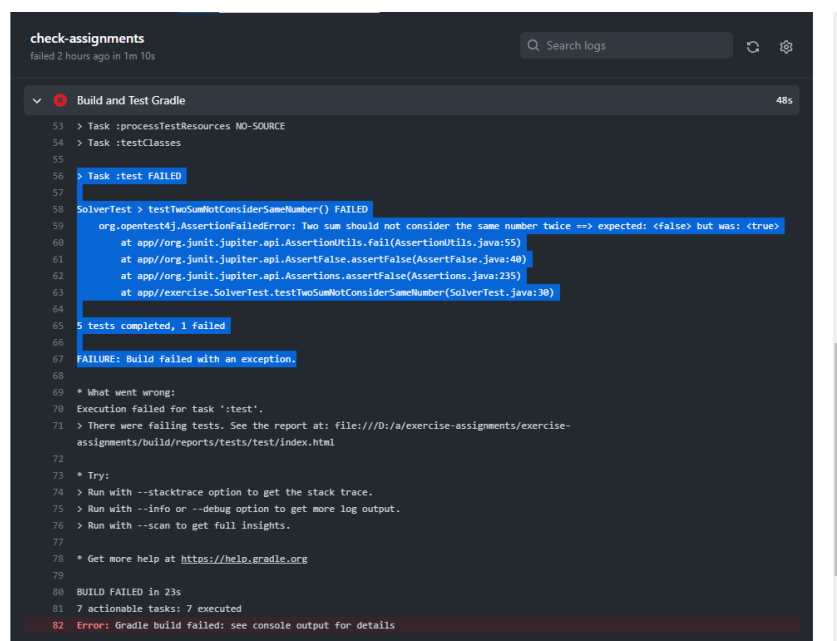
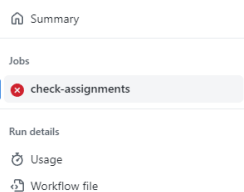
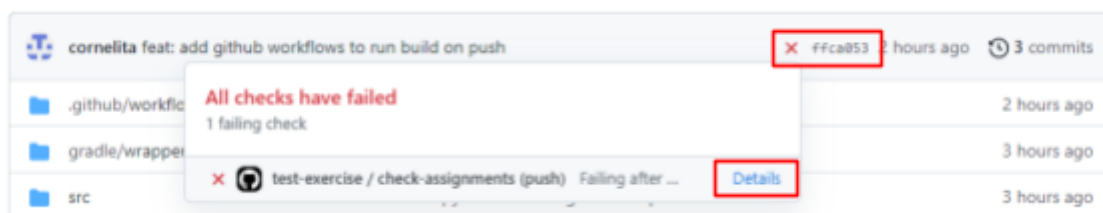
6. Unggah perubahan ke *repository*-mu yang berada di GitHub dengan menjalankan perintah: `git push origin master`. Jika diminta *username/email* dan *password*, gunakan detail akun GitHub-mu.

```
E:\ddp2\exercise-assignments>git push origin master
Enumerating objects: 13, done.
Counting objects: 100% (13/13), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (7/7), 521 bytes | 260.00 KiB/s, done.
Total 7 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/cornelita/exercise-assignments.git
ffca053..eb8ec12 master -> master
```

7. Setelah pembaruan diunggah, [GitHub Actions](#) akan menjalankan *build* termasuk *test*. Jika berhasil seperti di perangkatmu, maka akan ada centang hijau  pada *repository* GitHub **milikmu**.



8. Sebaliknya jika gagal, maka akan ada tanda silang merah. Untuk mengetahui apa yang menyebabkan kegagalan, klik tanda silang merah **✗** kemudian klik “Details”.

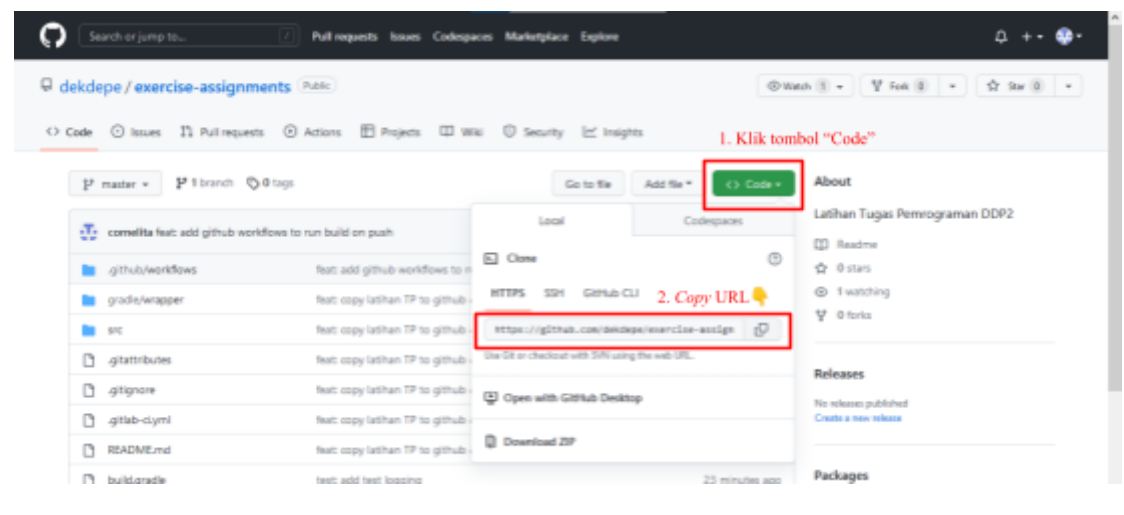


## F. Mendapatkan Pembaruan dari *Repository* dekdepe

Jika ada soal baru atau revisi soal, kamu bisa mendapat pembaruan dengan menambahkan repositori GitHub milik dekdepe sebagai *upstream*. Jalankan perintah

```
git remote add upstream <<url https repository milik dekdepe>>
```

**Note:** Cara mendapatkan URL HTTPS *repository* milik dekdepe jika lupa



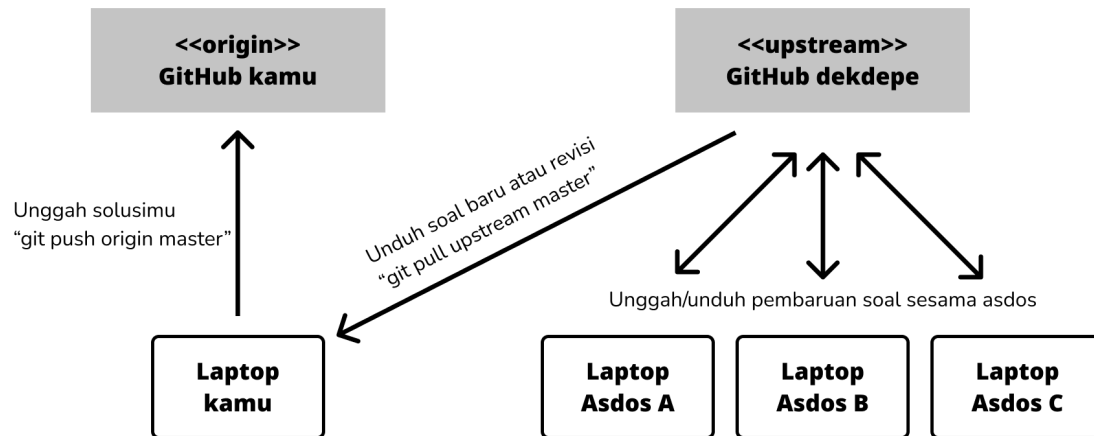
Pastikan *upstream* berhasil ditambahkan dengan menjalankan perintah: `git remote -v`.

```
E:\ddp2\exercise-assignments>git remote add upstream https://github.com/dekdepe/exercise-assignments.git
E:\ddp2\exercise-assignments>git remote -v
origin  https://github.com/cornelita/exercise-assignments.git (fetch)
origin  https://github.com/cornelita/exercise-assignments.git (push)
upstream https://github.com/dekdepe/exercise-assignments.git (fetch)
upstream https://github.com/dekdepe/exercise-assignments.git (push)
```

Nantinya, kamu bisa mendapat pembaruan soal atau revisi dengan menjalankan perintah

```
git pull upstream master
```

Berikut ini alur yang akan terjadi pada Tugas Pemrograman:



## Git sebagai Version Control System

1. Jalankan perintah `git log --oneline`.

```
E:\ddp2\exercise-assignments>git log --oneline
eb8ec12 (HEAD -> master, origin/master, origin/HEAD) fix: bug on isExistTwoSum
ffca053 feat: add github workflows to run build on push
9d500af test: add test logging
a038126 feat: copy latihan TP to github & update document
```

2. Pilih versi yang kamu inginkan dengan menyalin *commit hash*

```
E:\ddp2\exercise-assignments>git log --oneline
eb8ec12 (HEAD -> master, origin/master, origin/HEAD) fix: bug on isExistTwoSum
ffca053 feat: add github workflows to run build on push
9d500af test: add test logging
a038126 feat: copy latihan TP to github & update document
```

3. Keluar dari `git log` dengan menekan karakter `q` jika diperlukan
4. Jalankan `git checkout <commit_hash>` ke versi tersebut.

```
E:\ddp2\exercise-assignments>git checkout ffca053
Note: switching to 'ffca053'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:

    git switch -c <new-branch-name>

Or undo this operation with:

    git switch -

Turn off this advice by setting config variable advice.detachedHead to false

HEAD is now at ffca053 feat: add github workflows to run build on push
```

5. Lihat kembali file `src/main/java/exercise/Solver.java`. Perubahan yang telah kamu lakukan sebelumnya kini tidak ada.

```
public boolean isExistTwoSum(int[] nums, int K) {
    for (int i = 0; i < nums.length; i++) {
        for (int j = 0; j < nums.length; j++) {
            if (nums[i] + nums[j] == K) return true;
        }
    }
    return false;
}
```

6. Untuk kembali ke versi terbaru, jalankan perintah `git checkout master`.

```
E:\ddp2\exercise-assignments>git checkout master
Previous HEAD position was ffca053 feat: add github workflows to run build on push
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
```

7. Kini *bug fix* yang telah kamu kerjakan kembali ada.

```
public boolean isExistTwoSum(int[] nums, int K) {
    for (int i = 0; i < nums.length; i++) {
        for (int j = 0; j < nums.length; j++) {
            if (i != j && nums[i] + nums[j] == K) return true;
        }
    }
    return false;
}
```

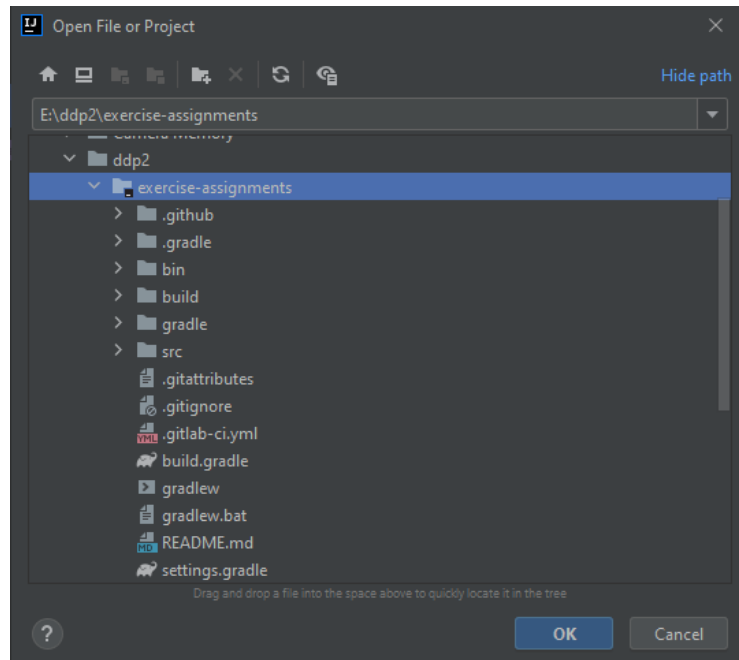
Sekarang, kamu telah dapat menggunakan *tools* yang dimanfaatkan pada tugas pemrograman! Sebenarnya, masih ada banyak kegunaan *tools* tersebut namun tidak dibahas karena tidak diperlukan dalam pengerjaan tugas pemrograman. Misalnya, `git branch` yang dimanfaatkan untuk kolaborasi sesama *developer* namun tidak diperlukan karena tugas pemrograman bersifat individu. Meski demikian, kamu disarankan mengeksplorasi sendiri kegunaan lain *tools* tersebut yang akan berguna dalam mata kuliah ataupun kegiatan lain seperti magang, proyek, dan pekerjaan di masa depanmu. Selamat mengeksplorasi!

## Additional Guide for IntelliJ

Untuk menginstall IntelliJ IDEA Ultimate, kalian dapat mengikuti panduan berikut

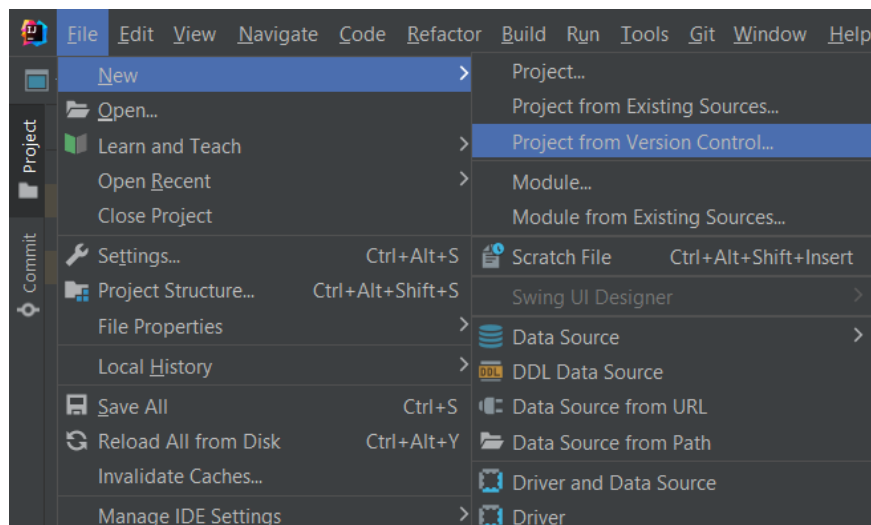
### Getting Started with IntelliJ IDEA Ultimate

Setelah menduplikat repository Latihan Tugas Pemrograman, kamu bisa meng-*import* project tersebut ke IntelliJ. Untuk melakukannya, kamu dapat membuka IntelliJ lalu pilih “Open”. Kemudian pilih project yang ingin di-*import*.



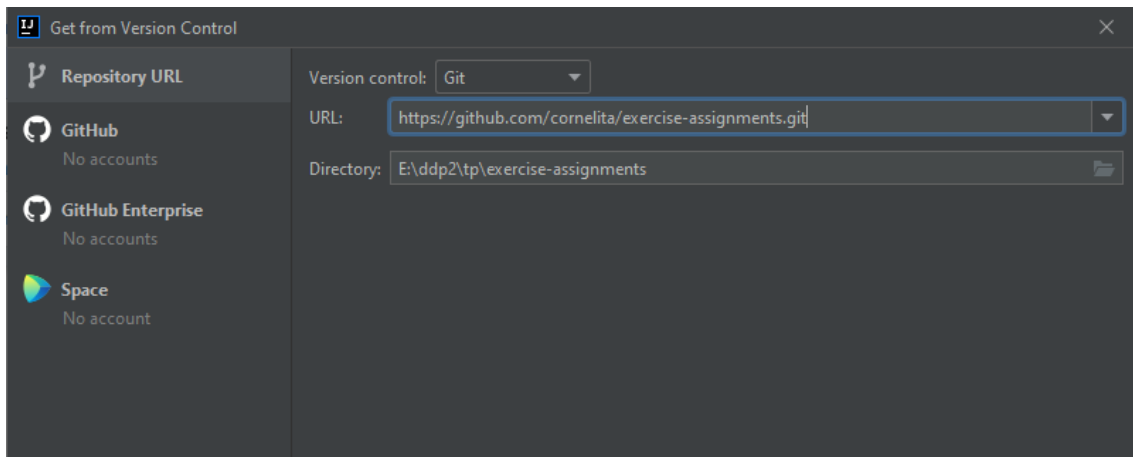
Alternatively, IntelliJ memiliki opsi untuk melakukan *clone*, membuat, dan melakukan *build* project langsung dari suatu remote Git repository.

1. Caranya adalah pilih “File -> New -> Project from Version Control...”

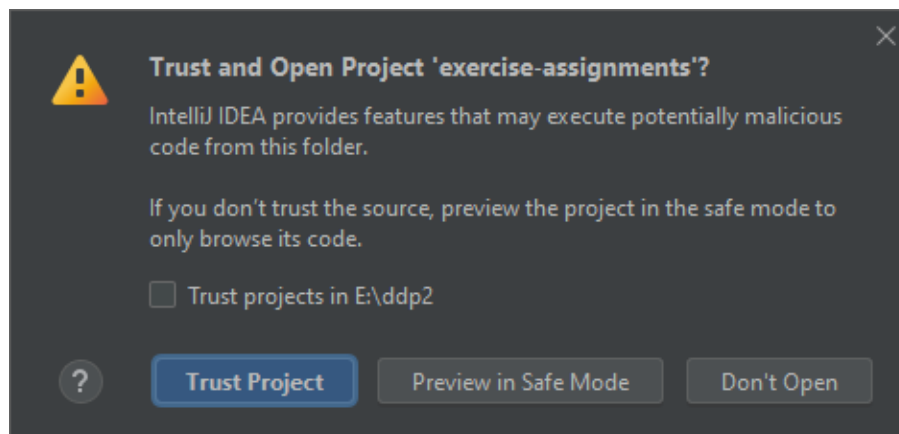




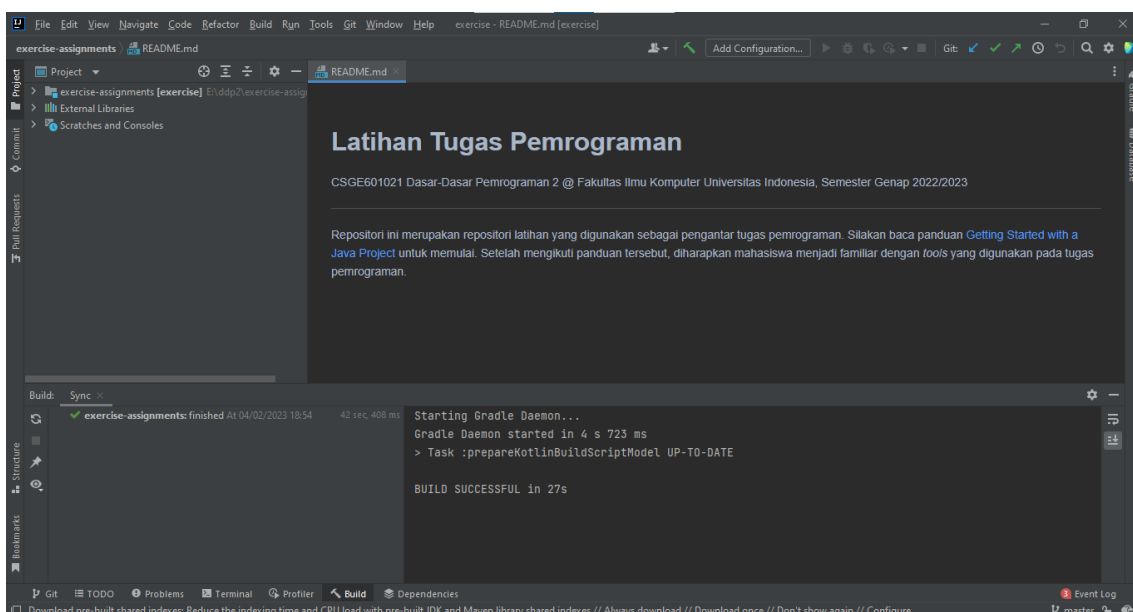
2. Masukkan URL HTTPS yang digunakan untuk clone, lalu tekan “Clone”.



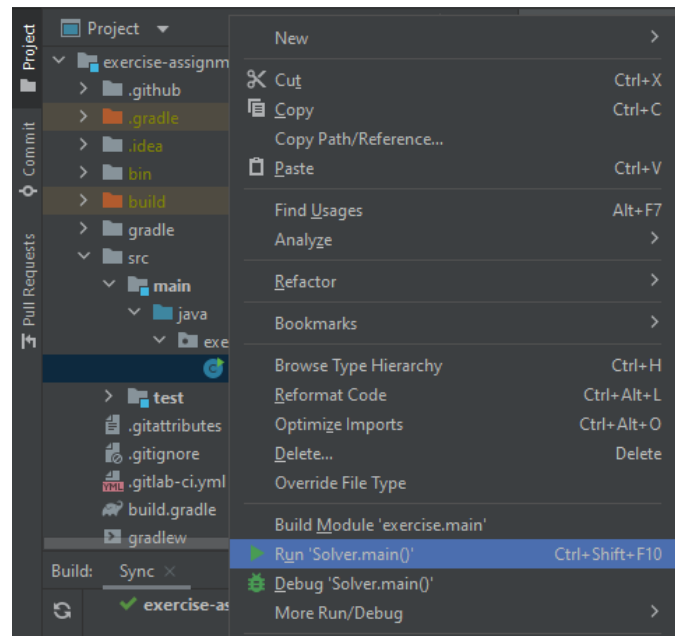
3. “Log In Via GitHub” jika diperlukan dan pilih “Trust Project” jika ditanya (aman koq)



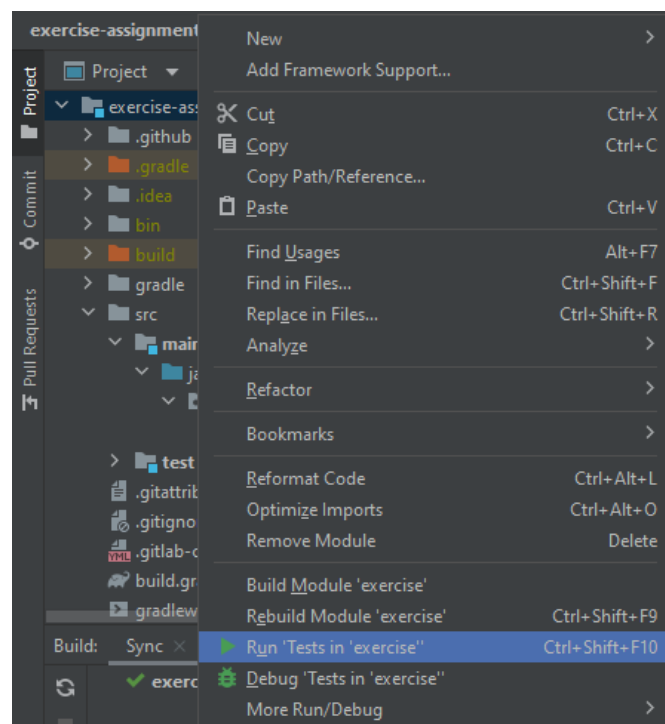
4. IntelliJ akan melakukan build Gradle project secara otomatis



5. Untuk menjalankan programnya, klik kanan pada **Solver.java** dan pilih **Run 'Solver.main()'**



6. Untuk menjalankan test, klik kanan pada folder **test/java** dan pilih **Run 'Tests in \'exercise.test\''**



7. Akan tampil seperti gambar dibawah ini (untuk kode yang belum diperbaiki)

```
Run: Tests in 'exercise'
Tests failed: 1, passed: 4 of 5 tests - 58 ms

Test Results
  exercise.SolverTest
    testTwoSumNotConsiderSameNumber

> Task :compileJava
> Task :processResources NO-SOURCE
> Task :classes
> Task :compileTestJava
> Task :processTestResources NO-SOURCE
> Task :testClasses
> Task :test

Two sum should not consider the same number twice ==> expected: <false> but was: <true>
Expected :false
Actual   :true
<Click to see difference>

org.opentest4j.AssertionFailedError: Two sum should not consider the same number twice ==> expected: <false> but was: <true>
    at app//org.junit.jupiter.api.AssertionUtils.fail(AssertionUtils.java:55)
    at app//org.junit.jupiter.api.AssertFalse.assertFalse(AssertFalse.java:40)
    at app//org.junit.jupiter.api.Assertions.assertFalse(Assertions.java:235)
    at app//exercise.SolverTest.testTwoSumNotConsiderSameNumber(SolverTest.java:30)
    at java.base@17.0.6/jdk.internal.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
    at java.base@17.0.6/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:77)
    at java.base@17.0.6/jdk.internal.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
    at app//org.junit.platform.commons.util.ReflectionUtils.invokeMethod(ReflectionUtils.java:725)
    at app//org.junit.jupiter.engine.execution.MethodInvocation.proceed(MethodInvocation.java:60)
    at app//org.junit.jupiter.engine.execution.InvocationInterceptorChain$ValidatingInvocation.proceed(InvocationInterceptorChain.java:131)
    at app//org.junit.jupiter.engine.execution.InvocationInterceptorChain.proceed(InvocationInterceptorChain.java:67)
    at app//org.junit.jupiter.engine.execution.InvocationInterceptorChain$ChainImpl.intercept(InvocationInterceptorChain.java:90)
```

Feel free to use, reuse, and share this work: the more we share, the more we have!



## Solusi Latihan Tugas Pemrograman

Untuk memperbaiki kode “isExistTwoSum”, tambahkan kondisi “i != j” seperti gambar di bawah ini.

```
public boolean isExistTwoSum(int[] nums, int K) {  
    for (int i = 0; i < nums.length; i++) {  
        for (int j = 0; j < nums.length; j++) {  
            if (i != j && nums[i] + nums[j] == K) return true;  
        }  
    }  
    return false;  
}
```