Files and input/output streams

File :تعد الملفات أهم آلية لتخزين البيانات بشكل دائم على أجهزة تخزين كبيرة السعة. يعني دائمًا عدم فقدان البيانات عند إيقاف تشغيل الجهاز. يمكن أن تحتوي الملفات على البيانات بتنسيق يمكن تفسيره بواسطة البرامج ، على سبيل المثال ، برنامج Java مخزن في ملف على القرص الصلب. في هذه الوحدة سنتعامل فقط مع الملفات النصية (**Text File**).

يتميز كل ملف باسم ودليل يتم وضع الملف فيه (يمكن للمرء أن يأخذ في الاعتبار المسار الكامل الذي يسمح له بالعثور على الملف على القرص الصلب كجزء من اسم الملف). أهم العمليات على الملفات هي: الإنشاء ، القراءة من ، الكتابة إلى.

Object Output/Input: يقوم بقراءة /كتابة أنواع البيانات الأولية وكائنات Java، يكون سهل الاستخدام اذا كنت تريد ارسال بيانات معقدة، و يعتبر Object اسرع مرتين تقريبا من Data لقراءة/كتابة مجموعة ضخمة من بيانات (huge array of double).

العمليات على الملفات

لتنفيذ عمليتي القراءة والكتابة على ملف ، يجب فتح الملف قبل القيام بالعمليات ، والإغلاق بعد أن ننتهي من تشغيله.

*فتح ملف يعني الإشارة إلى نظام التشغيل الذي نريد تشغيله على ملف من داخل برنامج Java ، ويقوم نظام التشغيل بالتحقق مما إذا كانت هذه العمليات ممكنة ومسموح بها. هناك طريقتان لفتح الملف: الفتح للقراءة والفتح للكتابة.

*إغلاق ملف يعني الإشارة إلى نظام التشغيل أن الملف الذي تم فتحه مسبقًا لم يعد مستخدمًا من قبل البرنامج. يؤدي إغلاق الملف أيضًا إلى ضمان نقل البيانات المكتوبة في الملف بشكل فعال إلى القرص الصلب.

Writing text files

لكتابة سلسلة نصية في ملف ، يتعين علينا القيام بما يلي:

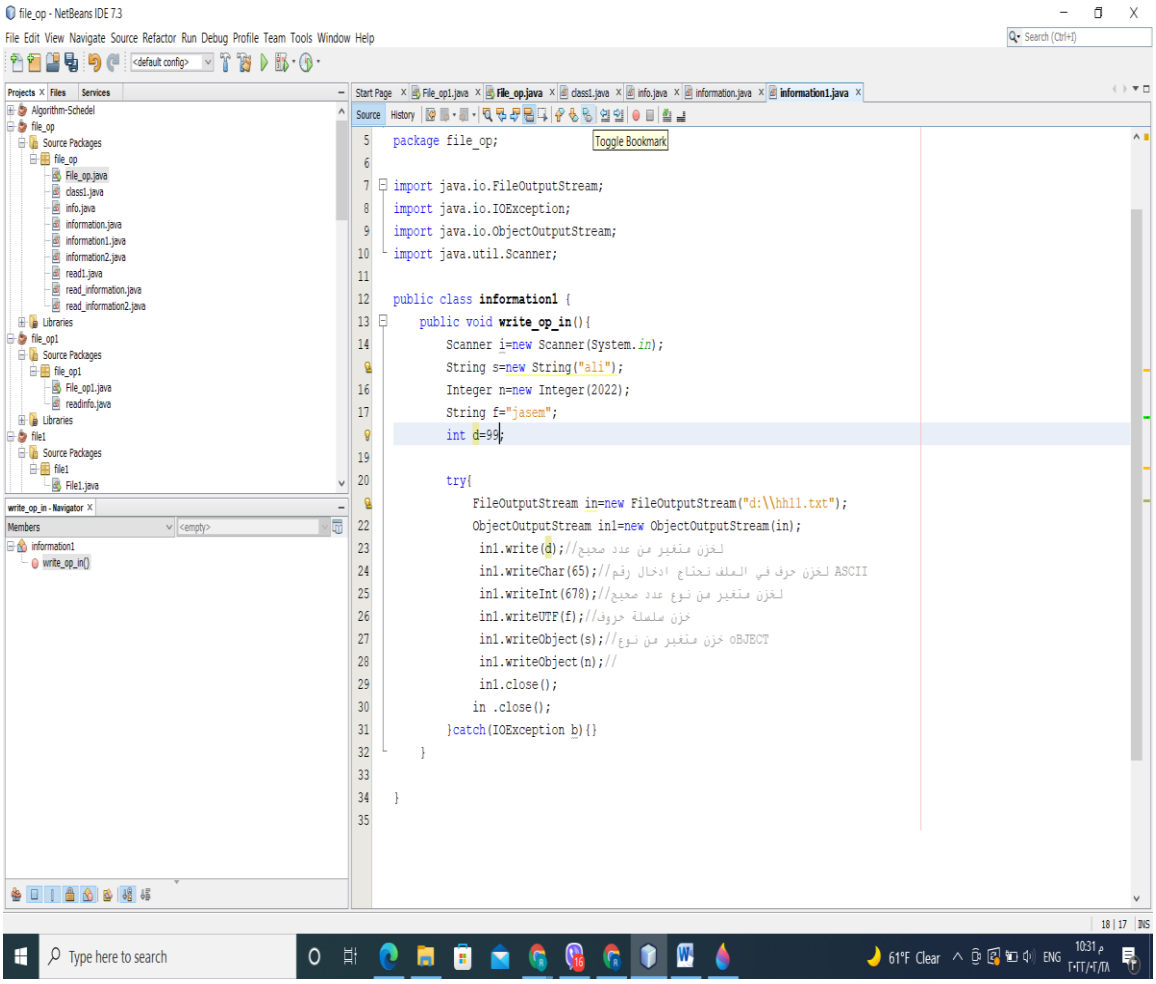
١. افتح الملف للكتابة عن طريق إنشاء كائن من الفئة FileOutputStream مرتبط باسم الملف . يتم استخدام فئة ObjectOutputStream لكتابة أنواع البيانات الأولية وكائنات Java (بيانات الاولية و بيانات نوع كائن)إلى OutputStream. يمكن فقط كتابة الكائنات التي تدعم واجهة java.io.Serializable في التدفقات.

٢. كتابة نص على الملف باستخدام الادوات.

٣. أغلق الملف عند الانتهاء من الكتابة إليه.

Program for writing to a file

تمثل الصورة رقم (١) برنامج لفتح ملف نصي من نوع(Text Decuments) على محرك الاقراص (D:\) وكتابة بيانات (البيانات الاعتيادية و بيانات من نوع Object) داخل ملف.



صورة (١)

Reading from a text file

لقراءة سلاسل نصية من ملف ، يتعين علينا:

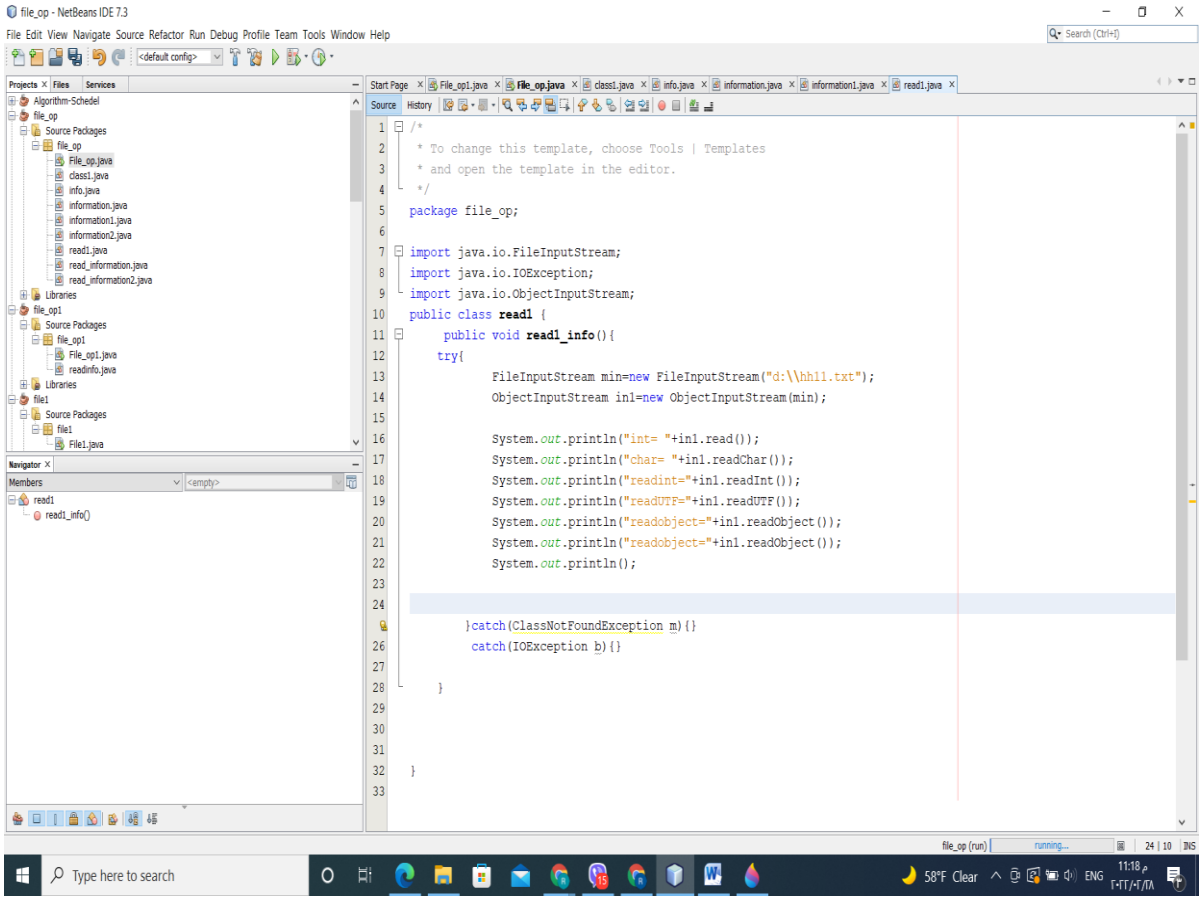
١. افتح الملف للقراءة عن طريق إنشاء كائن من فئة FileInputStream. يقوم ObjectInputStream بقراءة الكائنات والبيانات الأولية المكتوبة باستخدام ObjectOutputStream.

٢. قراءة سطور النص من الملف.

٣. أغلق الملف عند الانتهاء من القراءة منه.

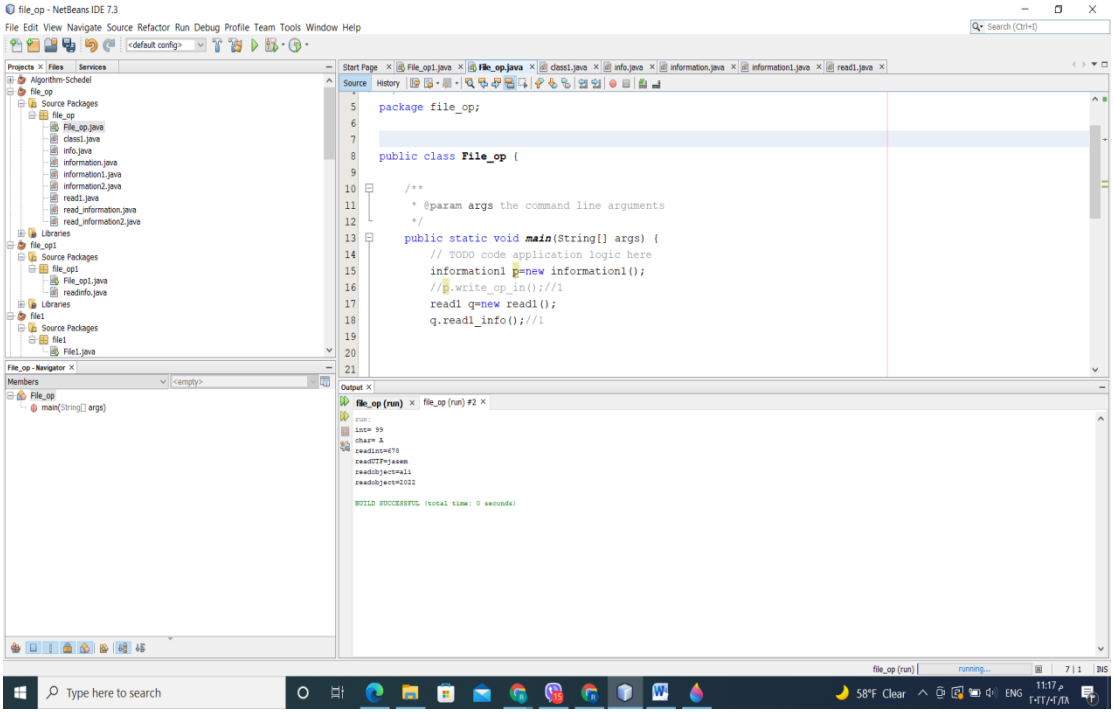
Program reading to a file

تمثل الصورة رقم (2) برنامج لفتح ملف من نوع(Text Decuments) على محرك الاقراص (D:\) وقراءة البيانات من داخل ملف.



صورة (٢)

تمثل الصورة رقم (٣) class mainحيث فقط تم استدعاء الدوال فيه و كذلك شاشة التنفيذ حيث تم طباعة البيانات التي تم سحبها من الملف على شاشة تنفيذ جافا.

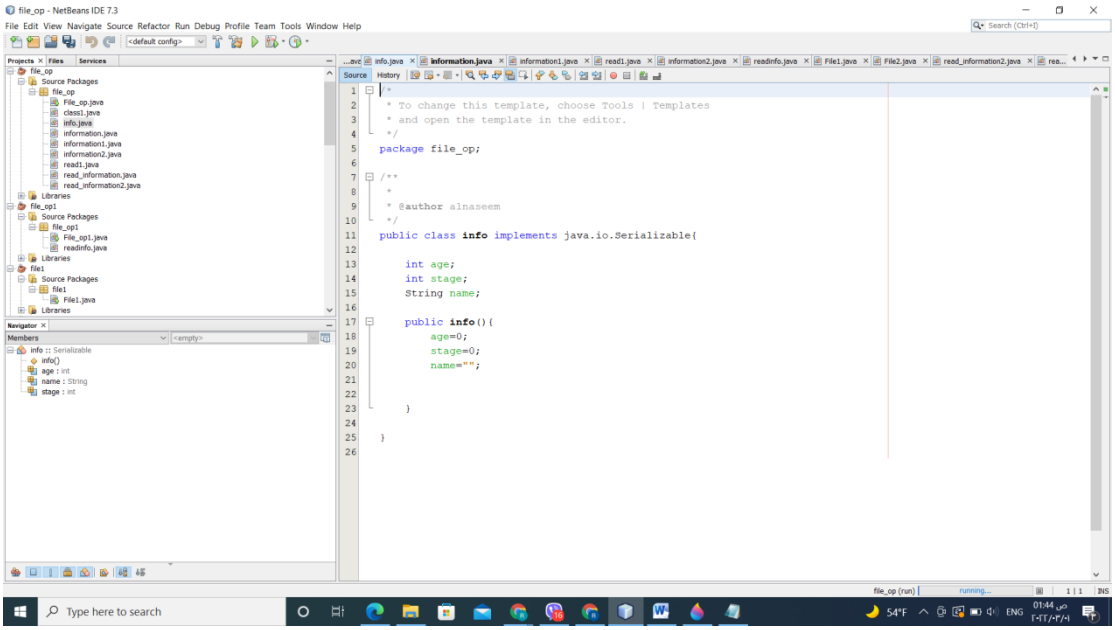


صورة (٣)

Example:-

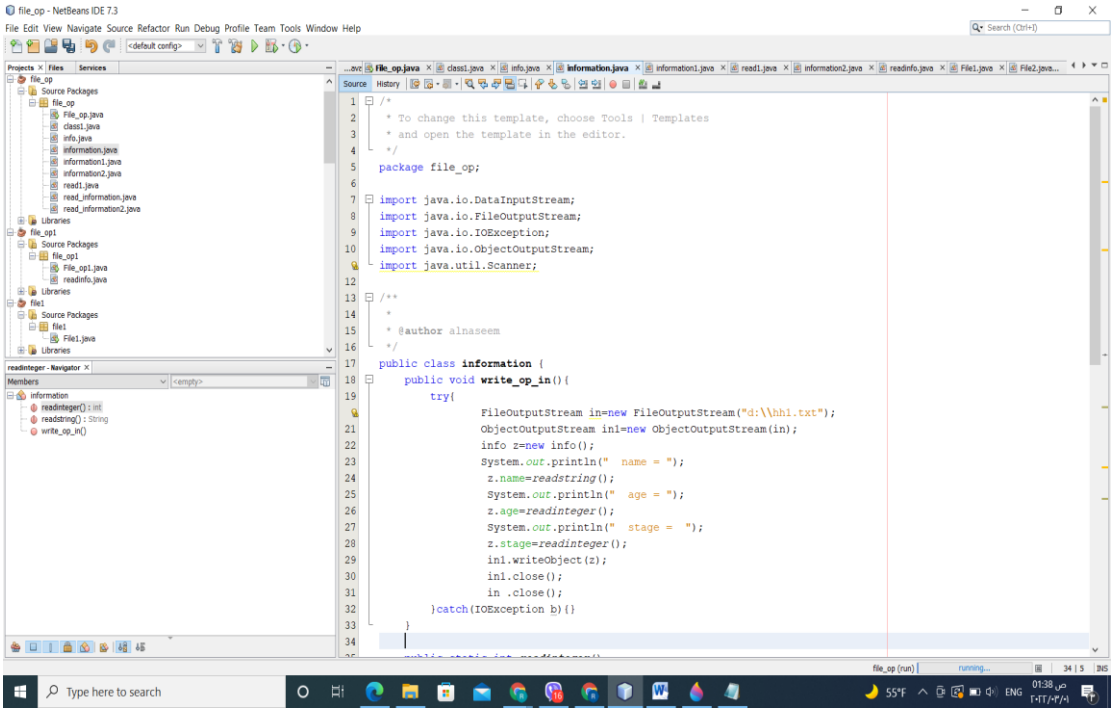
في هذا المثال نقوم بخزن بيانات طالب (اسم طالب ، عمر طالب، مرحلة) حيث يتم خزن هذه البيانات داخل فايل (Text Decument) و ذللك باستخدام متغيير واحد من نوع Object of class حيث يتم خلق sub class يتم تعريف بيانات طالب داخل sub classو يتم تضمينة في بيئة implements java.io.Serializable بعد اسم sub class.

تمثل صورة رقم (٤) خلق كلاس ثانوي sub class يتم تعريف البيانات مطلوبة داخل هذا الكلاس فقط (كلاس خاص بالتعريفات التي تمثل نوع الكائن) لتمثيل حالة طالب و هي اسم و عمر طالب و مرحلة. و من ثم كتابة دالة(implements java.io.Serializable) يكون بين اسم كلاس الثانوي و({}).



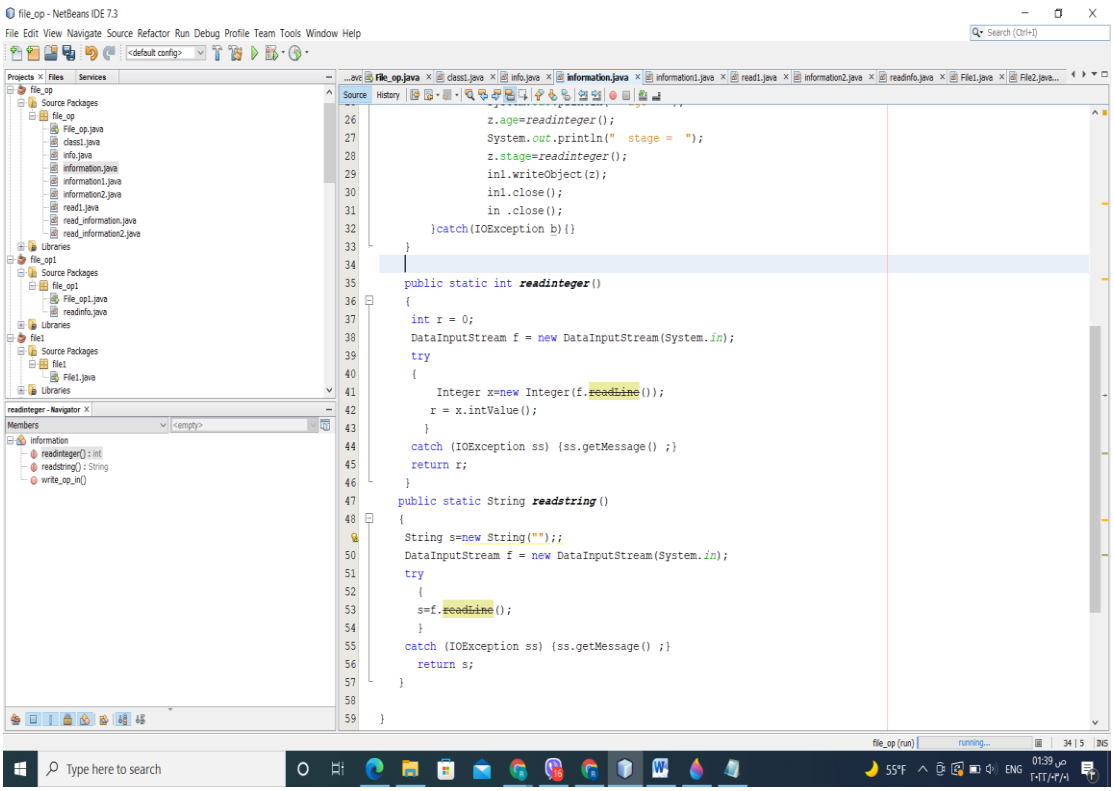
صورة (٤)

صورة رقم (٥) تمثل خلق sub class اخر لتضمين دالة الكتابة في الفايل باستخدام FileOutputStream



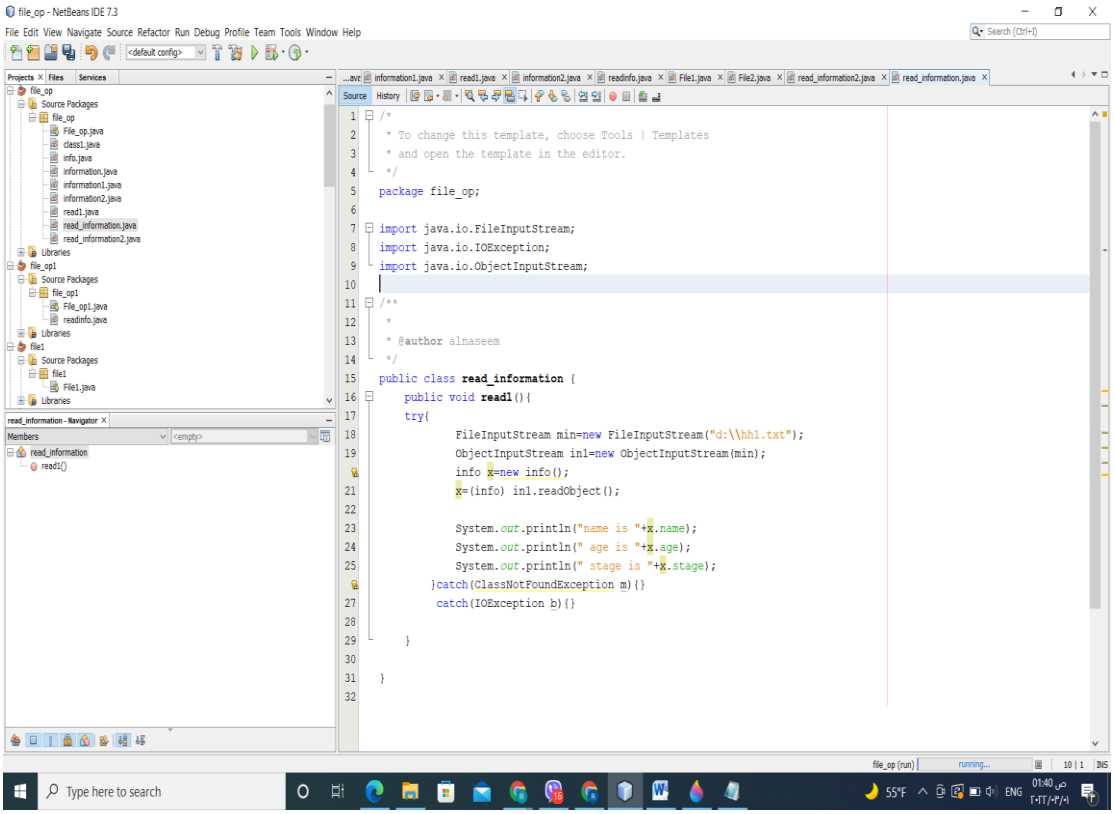
صورة (٥)

في هذا المثال صورة رقم (٦) تمثل دوال قراءة المتغييرات(اسم، عمر، مرحلة) من keyboard باستخدام دالة readinteger() لقراءة متغيير يمثل عدد صحيح، و دالة readstring() لقراءة متغيير يمثل سلسلة حروف.



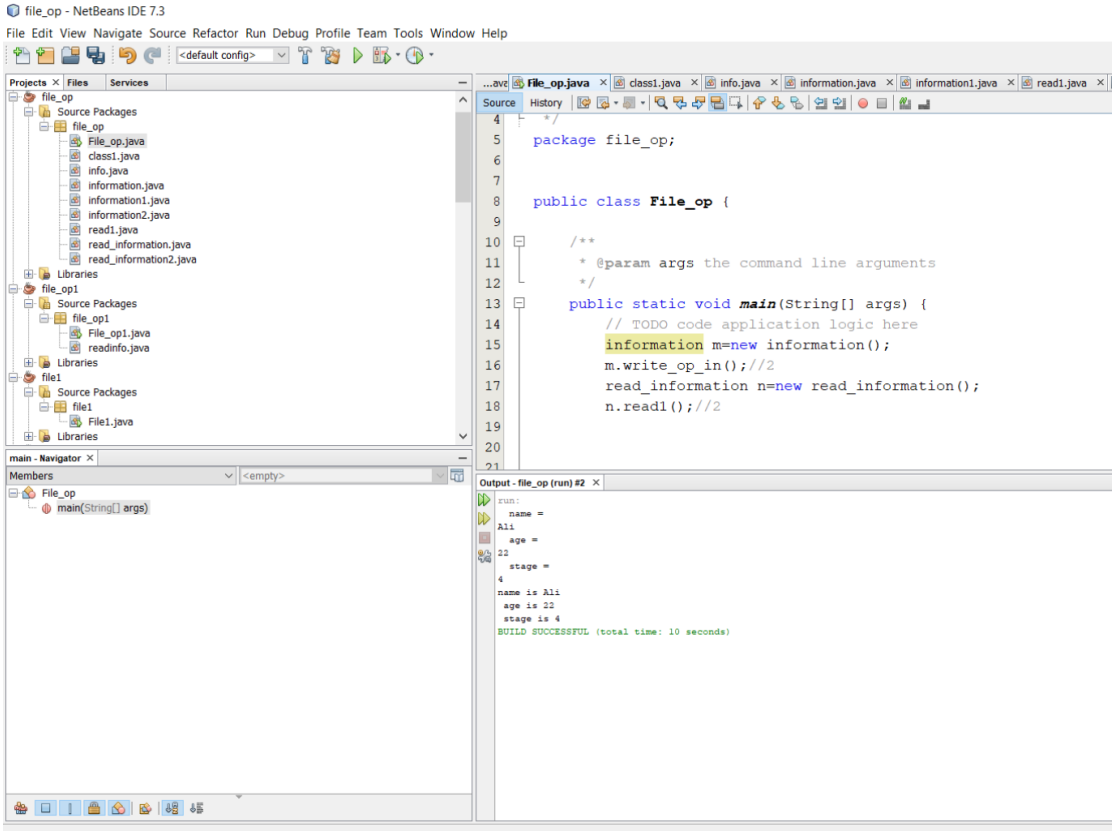
صورة (٦)

صورة رقم (٧) تمثل خلق sub class اخر لتضمين دالة قراءة بيانات من فايل.



صورة (٧)

صورة رقم (٨) تمثل main class حيث يتم فيه فقط استدعاء دالة كتابة بيانات في فايل و دالة قراءة بيانات من فايل، و شاشة تنفيذ للاسفل توضح حالة ادخال بيانات في حالة كتابة و طباعة بيانات بشاشة تنفيذ جافا عند قراءة.


file_op - NetBeans IDE 7.3
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Projects Files Services
file_op
Source Packages
file_op
File_op.java
class1.java
info.java
information.java
information1.java
information2.java
read1.java
read_information.java
read_information2.java
Libraries
file_op1
Source Packages
file_op1
File_op1.java
readinfo.java
Libraries
file1
Source Packages
file1
File1.java
Libraries
main - Navigator
Members
<empty>
File_op
main(String[] args)
File_op.java
class1.java
info.java
information.java
information1.java
read1.java
Source History
package file_op;
public class File_op {
/**
* @param args the command line arguments
*/
public static void main(String[] args) {
// TODO code application logic here
information m=new information();
m.write_op_in();//2
read_information n=new read_information();
n.read1();//2
Output - file_op (run) #2
run:
name =
Ali
age =
22
stage =
4
name is Ali
age is 22
stage is 4
BUILD SUCCESSFUL (total time: 10 seconds)

صورة (٨)