**پروژه نهایی کامپایلر – فاز اول (تحلیل­گر لغوی)**

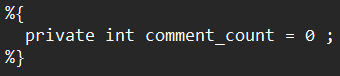
**بنیامین رمضانی (980122680008)**

**زهرا صداقت (990122681003)**

توضیحات کد **JFlex**:

با توجه به این که در فاز اول پروژ، یک تحلیل­گر لغوی (Lexical Analyzer) برای یک زبان شبه­جاوا مطلوب است که ابتدا عناصر کد و در ادامه قوانین مورد نظر (طبق توضیحات صورت پروژه)، بررسی خواهند شد.

در بخش اول کد، متغییر comment\_count را برای پیمایش بر روی کاراکترهای هر کامنت، با مقدار اولیه صفر تعریف می­کنیم.



سپس آپشن­های مورد نظر را به تناسب، اضافه می­کنیم:

Shape

Description automatically generated with medium confidence

**public:** نشان­دهنده نوع کلاس

**class subst:** نام کلاس که subst می­باشد.

**caseless و ignorecase:** برای حساس نبودن به حروف کوچک و بزرگ (Case Insensitive) طبق قانون اول

**standalone:** تولید فایل جاوایی با متد main که منجر به اجرای آن بدون کمک فایل دیگری می­شود.

**line:** برای دریافت و چاپ شماره خط

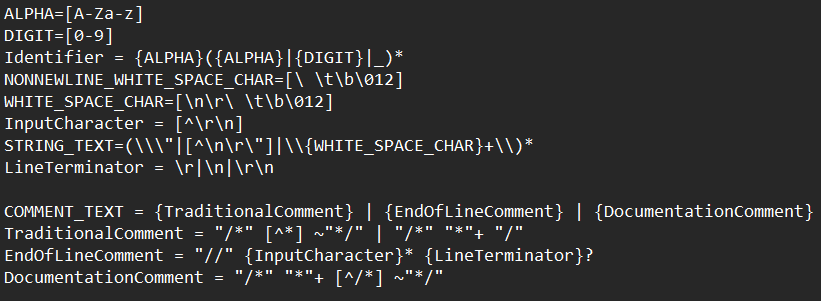
**char:** برای دریافت و چاپ کاراکترها

**column:** برای دریافت و چاپ شماره ستون

**state COMMENT:** برای شناسایی و چاپ اطلاعات مربوط به هر کامنت

**unicode:** فایل­هایی که توسط این اسکنر اسکن شوند، بر اساس Unicode خواهند بود. (از Unicode encoding استفاده می­کنند.)

در ادامه ماکروها و آپشن­های زیر را با فرمت­های خواسته شده تعریف می­کنیم:



**ALPHA:** تمام حروفی که توسط پارسر پذیرش می­شوند. این دسته­بندی شامل تمام حروف کوچک و بزرگ انگلیسی می­باشد.

**DIGIT:** تمام اعداد قابل پذیرش که ادامه به کمک این تعریف، اعداد صحیح و اعشاری تعریف خواهند شد.

**Identifier:** شناسه که نشان­دهنده هر فرمت ممکنی برای نام­گذاری متغیرها، کلاس­ها و... است.

**NONNEWLINE\_WHITE\_SPACE\_CHAR:** هر حالتی از کاراکتر­های فاصله، تب و... به جز خط جدید (\n)

**WHITE\_SPACE\_CHAR:** تمام کاراکتر­هایی که منجر به ایجاد فضای خالی می­شوند. (اجتماع ماکروی NONENEWLINE\_WHITE\_SPACE و InputChar)

**InputCharacter:** هر نوع کاراکتر به جز کاراکتر خط جدید

**STRING\_TEXT:** تمام حالت­هایی که برای پذیرش یک رشته مجاز هستند.

**LineTerminator:** حالت­هایی که نشان­دهنده انتقال پارسر به خط بعد هستند.

**COMMENT\_TEXT:** تعیین کننده تمام حالات قابل پذیرش برای کامنت که شامل TraditionalComment، EndOfLineComment و DocumentationComment می­باشد.

**TraditionalComment:** کامنت­های بلوکی جاوا که با /\* شروع شده و با \*/ به اتمام می­رسند.

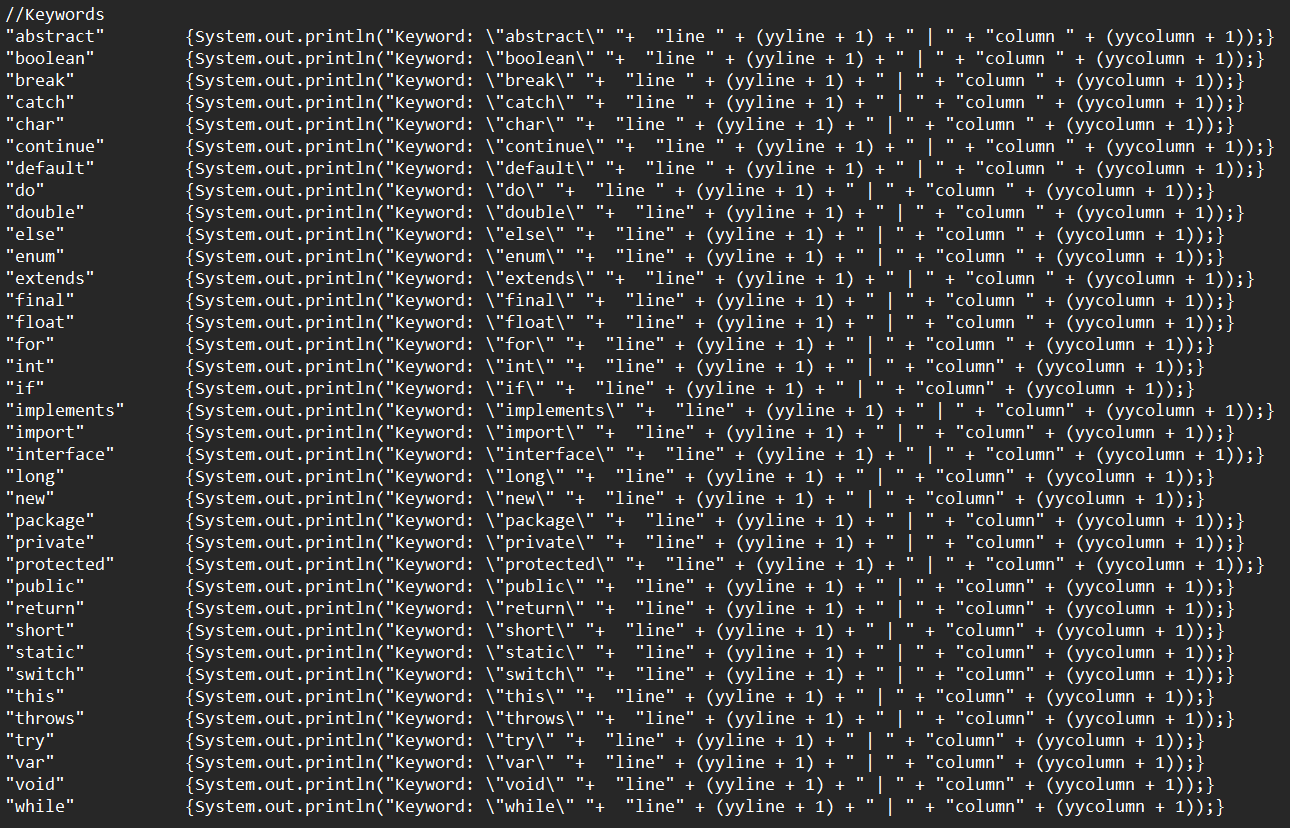
**EndOflineComment:** کامنت­های خطی جاوا که با // شروع می­شوند و با سر خط رفتن به اتمام می­رسند.

در بخش بعدی، از کد jflex که با <YYINITIAL> و براکت­های باز و بسته مشخص شده است، به تعریف انواع actionهای مرتبط با ماکروها و خواسته صورت پروژه می­پردازیم. هر تعریف و اکشنی در این بخش، داخل تعریف کلاس قرار می­گیرد.

کلیدواژه­ها (**Keywords**):

ابتدا کلیدواژه­های زبان مد نظر را تعریف می­کنیم و در بخش action هر کدام (طبق خواسته صورت پروژه)، شماره خط و ستون آن را در خروجی چاپ می­کنیم.

کلیدواژه­ها تعریف شده عبارتند از:

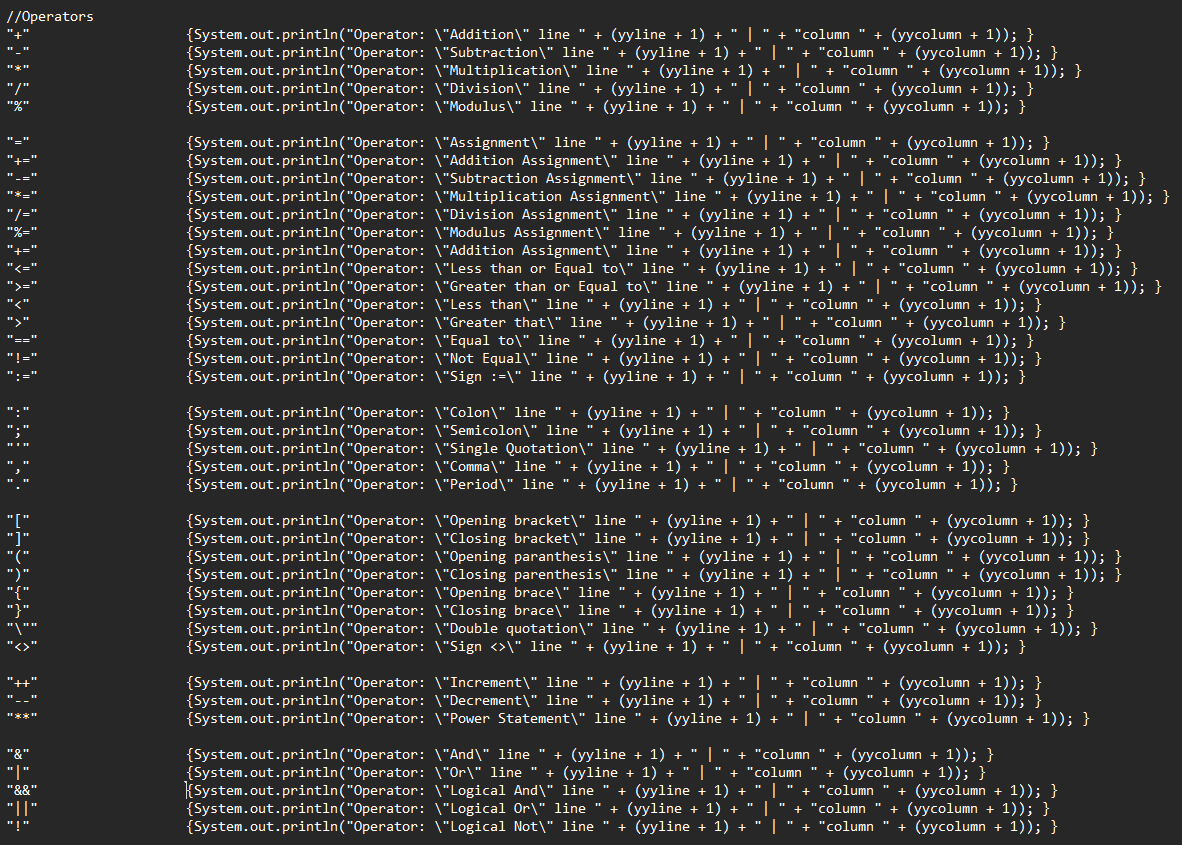


پس از شناسایی هر کلیدواژه، به فرمت زیر و با اطلاعات خوسته شده در خروجی چاپ می­شود:

Keyword: “Keyword Name” line x | column y

عملگرها (**Operators**):

عملگرها (Operators) را هم همانند کلیدواژه­ها تعریف می­کنیم:



پس از شناسایی هر عملگر، به فرمت زیر و با اطلاعات خوسته شده در خروجی چاپ می­شود:

Operator: “Operator Name” line x | column y

\*عملگر توان (POWER Statement) نیز در همین بخش تعریف شده است.

فضاهای خالی (**Whitespaces**):

برای شناسایی و چاپ خروجی در برخورد پارسر با هر کاراکتر خط جدید، فاصله، تب و... این action تعریف می­شود:

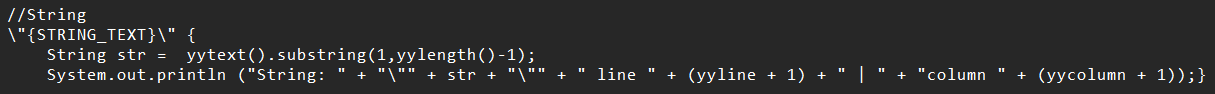


پس از شناسایی هر یک از کاراکترهای مرتبط، خطی با اطلاعات مذکور در خروجی چاپ خواهد شد:

Whitespace: line x |column y

رشته (**String**):

برای action رشته­های ورودی یا STRING\_TEXT خواهیم داشت:

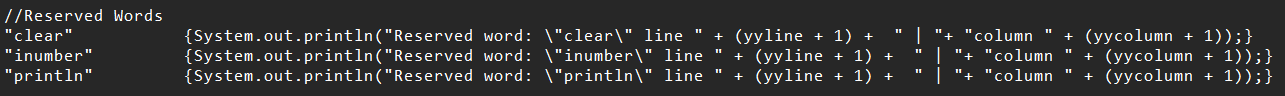


رشته­های شناسایی شده در ورودی به فرمت زیر در خروجی چاپ خواهند شد:

String: “String Name” line x | column y

کلمات رزرو شده (**Reserved Words**):

برای action کلمات رزرو شده یا Reserved Word، بنا بر سه کلمه خواسته شده CLEAR، inumber و println خواهیم داشت:



سه کلمه مذکورپس از شناسایی در ورودی، به فرمت زیر در خروجی چاپ خواهند شد:

Reserved Word: “Reserved Word Name” line x | column y

**شناسه­ها (Identifiers):**

برای action شناسه ها یا Identifiers خواهیم داشت:

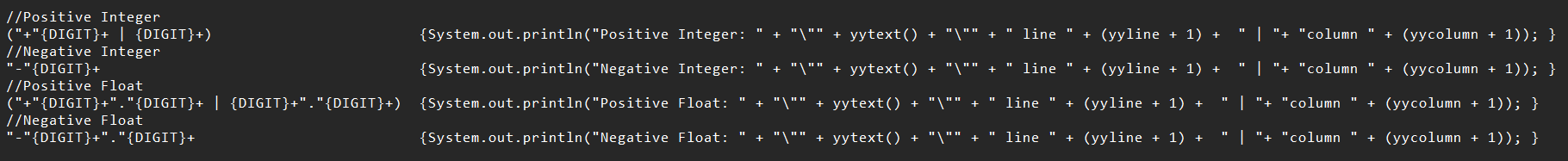


شناسه­های شناسایی شده در ورودی به فرمت زیر در خروجی چاپ خواهند شد:

Identifier: “Identifier Name” line x | column y

**اعداد صحیح و اعشاری منفی و مثبت:**

برای شناسایی اعداد صحیح و اعشاری منفی و مثبت، بدیهی است که اعداد منفی با علامت منفی و اعداد مثبت باید در دو حالت بدون علامت مثبت و با علامت مثبت شناسایی شوند.



اعداد صحیح مثبت شناسایی شده در ورودی به فرمت زیر در خروجی چاپ خواهند شد:

Positive Integer: “number” line x | column y

اعداد صحیح منفی شناسایی شده در ورودی به فرمت زیر در خروجی چاپ خواهند شد:

Nagative Integer: “number” line x | column y

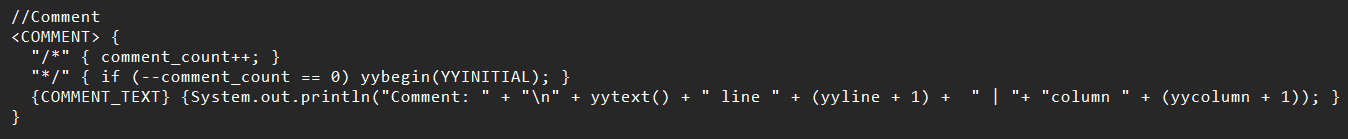
اعداد اعشاری مثبت شناسایی شده در ورودی به فرمت زیر در خروجی چاپ خواهند شد:

Positive Float: “number” line x | column y

اعداد اعشاری منفی شناسایی شده در ورودی به فرمت زیر در خروجی چاپ خواهند شد:

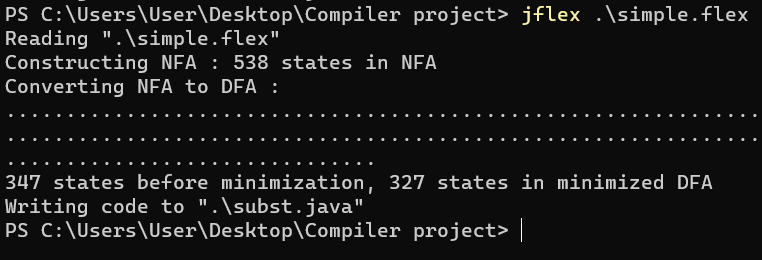
Nagative Float: “number” line x | column y

**کامنت­ها (Comments):**

کامنت­ها همانگونه که در ابتدای کد JFelx، تحت عنوان ماکرو و به فرمت کامنت­های زبان جاوا تعریف شده­اند، شناسایی خواهند شد:  


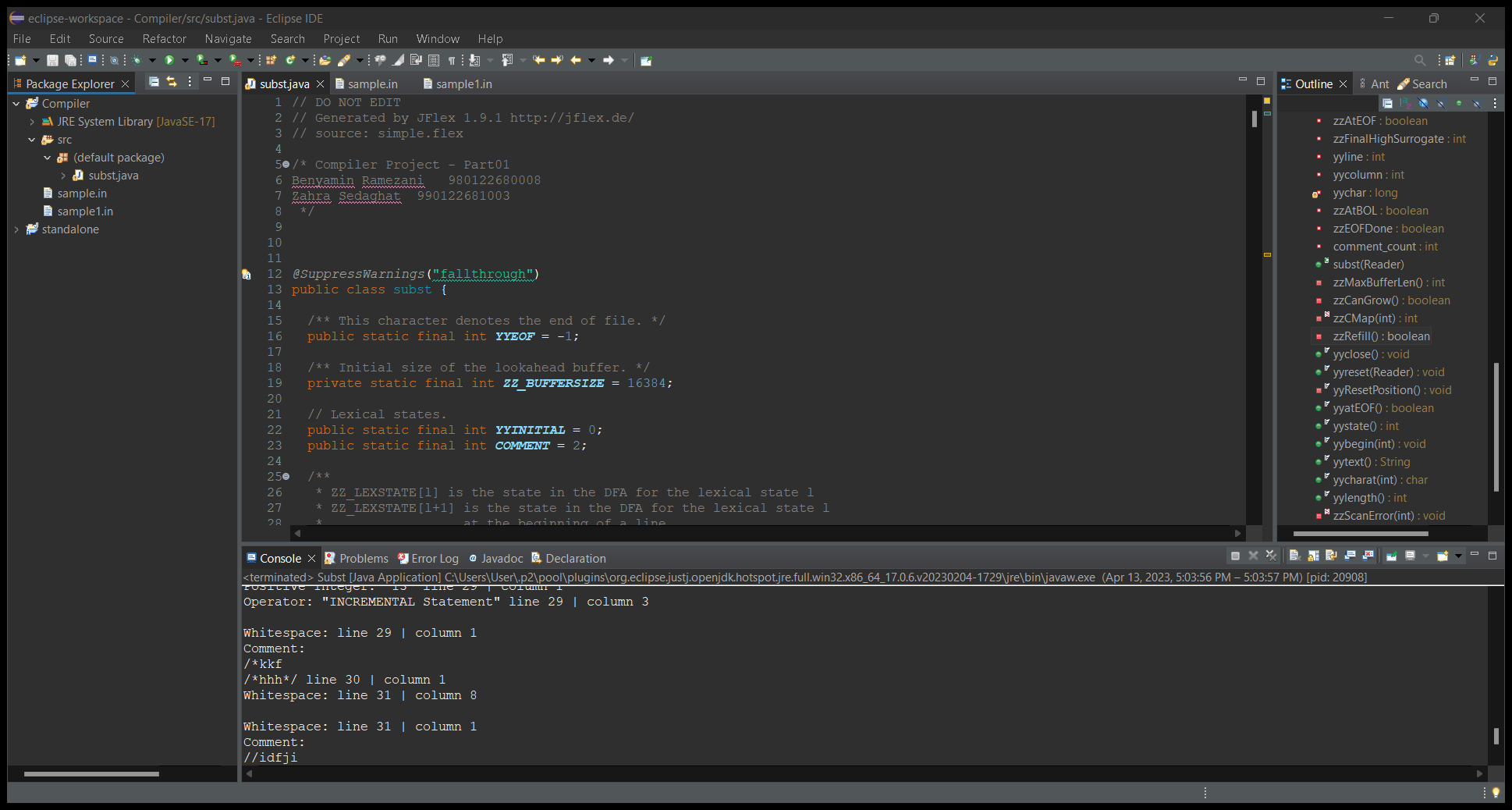
**نحوه اجرا:**

برای دریافت کد جاوای مربوطه، پس از نصب JFlex، ابتدا در cmd به دایرکتوری­ای که فایل simple.flex قرار دارد می­رویم و سپس با وارد کردن دستورات مربوطه (به شکل زیر) خروجی subst.java تولید می­شود.

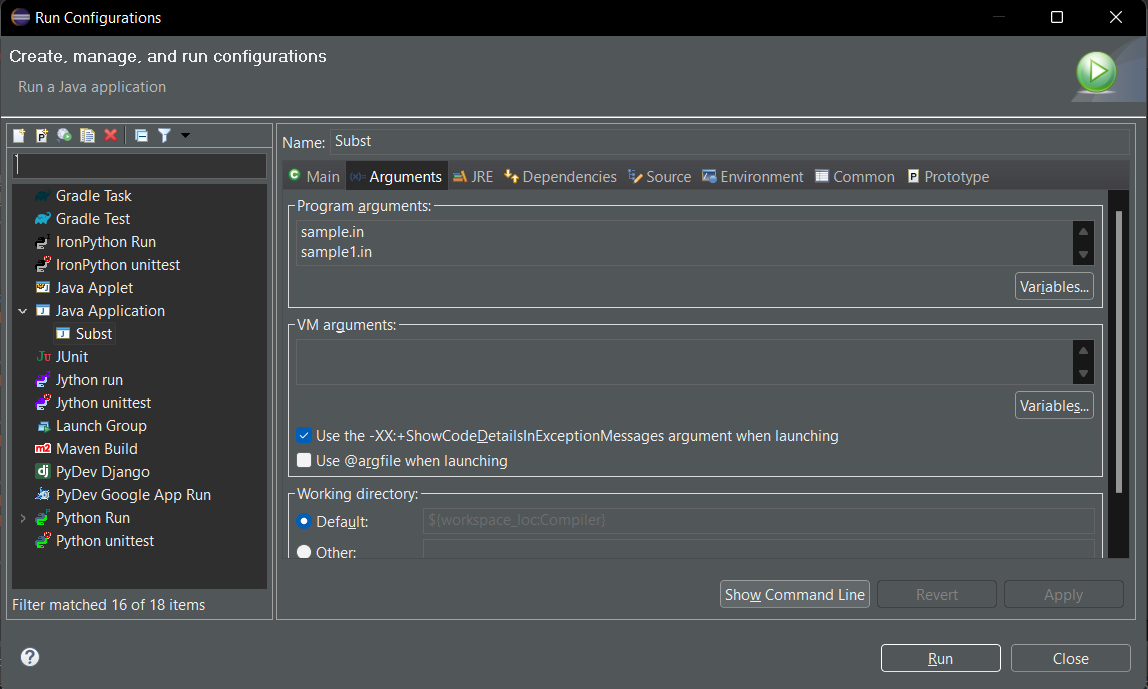


سپس فایل subst.java که طی مراحل مذکور تولید شده را در Eclipse باز می­کنیم. در این فایل کد جاوای زبان شبه­جاوایی که طی دستورات مراحل قبل تعریف کرده­ایم را خواهیم داشت.

برای تست این کد یک یا چند فایل sample ایجاد می­کنیم و اسامی فایل یا فایل­های تولید شده را در قسمت Run Configuration و سپس بخش argument مربوط به subst.java اضافه می­کنیم. با این کار، پس از هر بار اجرای این کد، سمپل یا سمپل­های تعریف شده به عنوان ورودی به کد داده خواهند شد و خروجی متناسب با محتوای هر سمپل را مشاهده خواهیم کرد.



\*در این جا دو فایل سمپل برای پروژه به نام­های sample.in و sample1.in تعریف شده­اند.



\*همان طور که مشخص است، نام هر دو فایل به عنوان آرگیومنت­های subst.java ذکر شده­اند.

محتوای هر سمپل و خروجی تولید شده از هر کدام در پوشه پروژه وجود دارند.