

## Project One Specification – Gene Models

Starting with Project One, document modules, methods and instance variables. See **Week 7, Lecture 1 Documentation** for details.

### Part A. Setting up

Use `svn mkdir` to create a new directory called “project1” to your section of the class repository. Use `svn cp` to make a copy of your `hw7_objects.py` object, rename it `p1.py`, and save it into your project1 directory. (Don’t just copy and paste the file and then add it – use `svn cp` to allow retrieval of earlier versions before the name and location change.)

### Part B. UML class diagram.

In this project, you will create a small class hierarchy involving three classes: `Range` (from HW7), `DNAFeature`, and `GeneModel`. Before you begin coding, sketch a UML class diagram showing class names, instance variables, method names, and relationships between classes. As you proceed in developing your code, update and revise your diagram as needed. You should use your class diagram as a blueprint of your code. For full credit, turn in your revised, updated class diagram on the project due date. See **Week 5, Class 2 OOP – Inheritance, UML**.

### Part C. Type and value checking.

Modify your `Range` class so that it generates appropriate `ValueError` and `TypeError` exceptions when users pass incorrect types and values to the `Range` class constructor and setter/getter methods. See **Week 6, Lecture 1 Errors and Exceptions**.

### Part D. DNAFeature - Subclass Range

Create a subclass of `Range`, called `DNAFeature`, that represents a `Range` that also has a strand and a sequence name. Implement “`setStrand`” and “`getStrand`,” which set and return strand information, and `setSeqName` and `getSeqName`, which set or return the name of the sequence the feature belongs to.

If a feature is on the minus (reverse) strand, `getStrand()` should return -1. If a feature is on the plus strand, `getStrand()` should return 1. If strand is not set, `getStrand()` should return 0. As with all get/set methods, they should return/accept the same argument types and values.

**Note:** In this and other subclasses, you are welcome to implement as many methods as you like in addition to the ones described in the specification.

### Part E. GeneModel

Create a new class – `GeneModel` - that contains a group of `DNAFeature` objects representing exons and is a child class of `DNAFeature`. It should implement the following methods:

`getFeats()` – returns a list of `DNAFeature` objects, sorted by start position  
`addFeat(feats)` – accepts a `DNAFeature` `feat` and adds it to its internal group of `DNAFeature` objects

`setTranslStart(i)` – accepts a non-negative int, sets the start position of the translated region.

`getTranslStart()` – returns an int, the start position of the translated region (`thickStart`)

`setTranslStop(i)` – accepts a positive int, sets the end position of the translated region (`thickStop`)

`getTranslStop()` – returns an int, the end position of the translated region

`setDisplayId(s)` – sets the name of the gene model; `s` is a string

`getDisplayId()` – return the name of the gene model, returns a string, e.g., AT1G10555.1

`GeneModel` should raise appropriate `ValueError` and `TypeError` exceptions when users pass incorrect types and values to constructors and “set” methods.

### **Part F. Bed.py**

Write a new module called `Bed.py` that has the following methods:

`readBed(file)` – read a BED format file and constructs a list of gene model objects from the data it contains.

`writeBed(models=models, fname=file)` – writes the given list of gene model objects and writes them to a file named `fname`.

### **Part G. findNmdTargets.py (Optional for now, solve this for HW10)**

Nonsense-mediated decay is a form of RNA quality control in which spliced mRNA transcripts containing premature stop codons (PTCs) resulting from aberrant splicing are targeted for degradation. Current thinking is that if a stop codon is located more than 50 bases upstream (5-prime) of an exon-exon junction in a spliced transcript, then the transcript will be susceptible to degradation by the NMD pathway. The proposed mechanism for NMD is that proteins involved in splicing remain attached to the newly-synthesized mRNA even after it is exported from the nucleus – these proteins form the exon junction complex, or EJC. Unless the ribosome displaces the EJC during translation, the EJC can stimulate degradation of the mRNA by the NMD pathway. It has been proposed that the ribosome can displace an EJC provided the EJC is located no more than 50 bases 3 prime of the termination codon.

The degree to which NMD pathways operate in plant cells is not well known. In the reference plant *Arabidopsis thaliana*, some annotated gene models contain stop codons located upstream of exon-exon junctions, but it is not clear if the corresponding spliced mRNA transcripts undergo NMD. Thanks to new sequencing-based expression analysis techniques, it is now much easier to assess the degree to which NMD occurs in plant cells by assessing the number of sequence reads derived from NMD-susceptible transcripts. If NMD is widespread, then transcripts containing termination codons sufficiently upstream of exon-exon junctions should appear rarely in RNA-Seq data sets. However, to assess this, we first must identify annotated gene models that contain stop codons more than 50 bases upstream of an exon-exon junction.

Write a program `findNmdTargets.py` that accepts two arguments: the name of a BED format file and an integer `N` and identifies all the gene models with exon-exon junctions `N` or more bases downstream of the annotated stop codon.

Use your program to identify all potential nonsense-mediated decay targets in this BED format file:

[http://bioviz.org/quickload/A\\_thaliana\\_Jun\\_2009/TAIR9\\_mRNA.bed.gz](http://bioviz.org/quickload/A_thaliana_Jun_2009/TAIR9_mRNA.bed.gz)

**BED format description:**

[http://teaching.transvar.org/6111/gene\\_models.pdf](http://teaching.transvar.org/6111/gene_models.pdf)