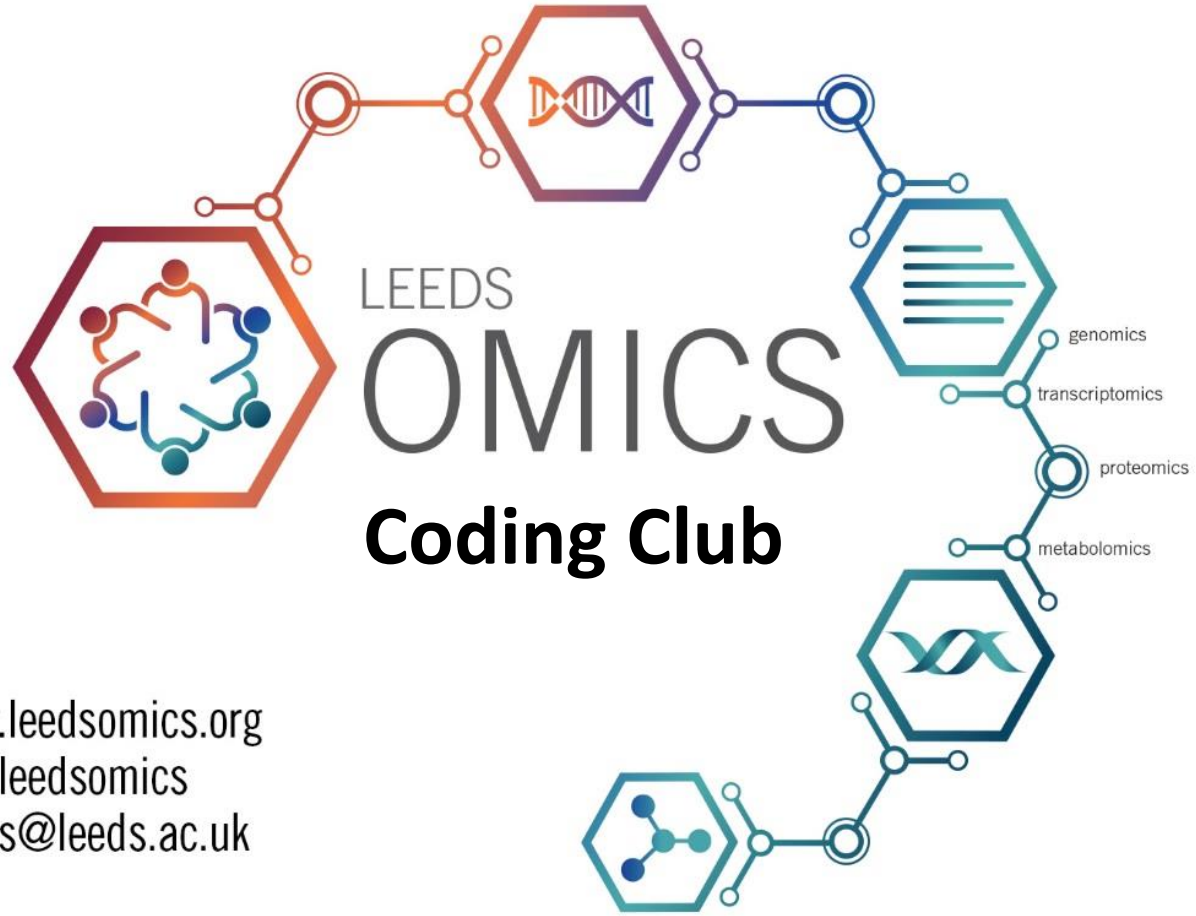


Coding Club

www.leedsomics.org
@leedsomics
omics@leeds.ac.uk

Good Practices in Writing Code: variable types

Club Moderators: Elton Vasconcelos, Euan McDonell, Chew Cheng, and Dapeng Wang

Topics to be addressed on the **2020-21** season - **Survey Result**

1st, 4th, 6th, 8th, ... sessions

2nd session

3rd, 5th, 7th, 9th, ... sessions

1 Which of the following topics would you like to attend in our Coding Club sessions?

LEEDS OMICS

genomics
transcriptomics
proteomics
metabolomics

Coding Club

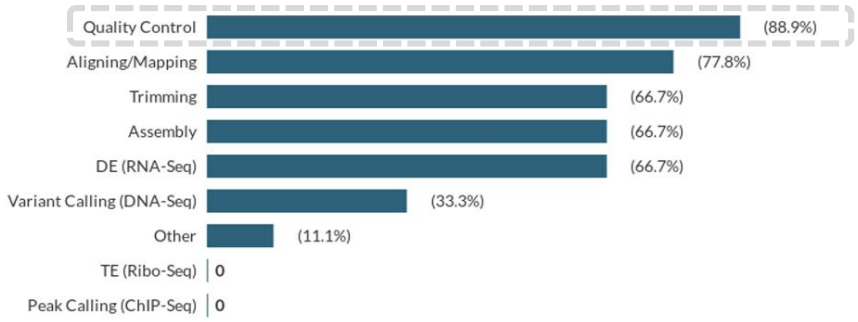
Survey Result (2020-2021)

www.leedsomics.org
@leedsomics
omics@leeds.ac.uk

1.b Do you think we should address "Good practices in writing code" topic in more sessions?

1.c Do you think we should address "Dealing with NGS data" topic in more sessions?

1.c.i Which sub-topics would be of most interest to you?



1st session (15/06/20)

- **Variable in programming** → programmer-predefined entity that will store information in the code

Different types of variables/data in the three main languages used in Bioinformatics

PERL

- Scalars ($)
- Arrays (@)
- Hashes (%)

https://www.tutorialspoint.com/perl/perl_variables.htm

Python

- Numbers
- String
- List
- Tuple
- Dictionary

https://www.tutorialspoint.com/python3/python_variable_types.htm

R

- Numeric
- Character
- Vectors
- Factors
- Data Frames

https://www.tutorialspoint.com/r/r_data_types.htm

NOTE: List and Array terminologies in R have different and more complex concepts than in PERL and Python.

Examples in PERL

- **Scalars**

Live Demo

```
#!/usr/bin/perl

$age = 25;              # An integer assignment
$name = "John Paul";    # A string
$salary = 1445.50;      # A floating point

print "Age = $age\n";
print "Name = $name\n";
print "Salary = $salary\n";
```

This will produce the following result −

```
Age = 25
Name = John Paul
Salary = 1445.5
```

https://www.tutorialspoint.com/perl/perl_variables.htm

Examples in PERL

- **Arrays**

Live Demo

```
#!/usr/bin/perl

@ages = (25, 30, 40);
@names = ("John Paul", "Lisa", "Kumar");

print "\$ages[0] = $ages[0]\n";
print "\$ages[1] = $ages[1]\n";
print "\$ages[2] = $ages[2]\n";
print "\$names[0] = $names[0]\n";
print "\$names[1] = $names[1]\n";
print "\$names[2] = $names[2]\n";
```

Here we used escape sign (\) before the $ sign just to print it. Other Perl will understand it as a variable and will print its value. When executed, this will produce the following result –

```
$ages[0] = 25
$ages[1] = 30
$ages[2] = 40
$names[0] = John Paul
$names[1] = Lisa
$names[2] = Kumar
```

https://www.tutorialspoint.com/perl/perl_variables.htm

Examples in PERL

- **Hash table**

Live Demo

```
#!/usr/bin/perl

%data = ('John Paul', 45, 'Lisa', 30, 'Kumar', 40);

print "\$data{'John Paul'} = $data{'John Paul'}\n";
print "\$data{'Lisa'} = $data{'Lisa'}\n";
print "\$data{'Kumar'} = $data{'Kumar'}\n";
```

This will produce the following result –

```
$data{'John Paul'} = 45
$data{'Lisa'} = 30
$data{'Kumar'} = 40
```

https://www.tutorialspoint.com/perl/perl_variables.htm

→ Useful short code that employs all three PERL variable types in order to combine information from two files

```
#!/usr/bin/perl
open(FILE1, "x.tsv");
open(FILE2, “y.tsv");
while(<FILE2>) {
        chomp($_);
        @array2 = split(/\t/, $_);
        $hash{$array2[0]} = $array2[2];
}
while(<FILE1>) {
        chomp($_);
        @array1 = split(/\t/, $_);
        if($hash{$array1[0]} ne "") {
                print("$_\t$hash{$array1[0]}\n");
        }
        else {
                print("$_\tNA\n");
        }
}
```

FILE1 x.tsv → a typical DE analysis output table with 3125 rows

Long_geneID	logFC.2h	logFC.17h	logCPM	LR	PValue	FDR
NC_018726.3_mrna_XM_023252992.1_17213	-0.0115539525378653	12.3552392903438	2.86329600280231	153.177804483085	5.46842652825283e-34	3.97729598252885e-29
NC_018737.3_mrna_XM_023245341.1_62253	-2.41190133112974e-08	10.7952771226017	1.31660408910246	123.483120559081	1.53455008109868e-27	5.10785193903218e-23
NC_018729.3_mrna_XM_023256742.1_28810	0.000496516357557754	10.763096713483	1.28333375331224	122.849205260556	2.10685197947211e-27	5.10785193903218e-23
NC_018736.3_mrna_XM_019818127.2_58617	-9.33346500212488	-10.032294222076	0.293441063845244	111.291091024287	6.81469842628745e-25	1.23911661485185e-20
NC_018736.3_mrna_XM_019818041.1_58415	-8.34999208356182	-9.51603983117479	0.890988635810215	108.822101347149	2.34196789867635e-24	3.40672018413056e-20
NC_018731.3_mrna_XM_019839937.2_40571	-9.53613707062572	-9.53619935953011	-0.197449535373358	105.301303647077	1.36179722364347e-23	1.65077059450061e-19
NC_018735.3_mrna_XM_023242979.1_55286	1.31546722469771	9.94937447648528	0.58790776418027	100.232406950434	1.71715451761346e-22	1.78417260535803e-18
NC_018741.3_mrna_XM_023249335.1_70881	0.450036882025638	9.6533406059691	0.228666387292216	99.2434452989686	2.81552683373531e-22	2.55973622089046e-18
NC_018724.3_mrna_XM_019822718.2_7002	3.71124136142327e-08	9.38866457539553	-0.0287171084086663	97.1391471277462	8.06310309653979e-22	6.51606238241702e-18

FILE2 y.tsv → gene product descriptions with 72,732 lines

NC_018723.3_ncrna_XR_002147070.2_1	XR_002147070.2	uncharacterized LOC102899061, transcript variant X3
NC_018723.3_ncrna_XR_002147044.2_2	XR_002147044.2	uncharacterized LOC102899061, transcript variant X2
NC_018723.3_ncrna_XR_002147038.2_3	XR_002147038.2	uncharacterized LOC102899061, transcript variant X1
NC_018723.3_mrna_XM_023247174.1_4	XM_023247174.1	peptidoglycan recognition protein 4
NC_018723.3_trna_5	5	tRNA-Glu
NC_018723.3_mrna_XM_011289120.3_6	XM_011289120.3	coiled-coil domain containing 115, transcript variant X1
NC_018723.3_mrna_XM_003980242.5_7	XM_003980242.5	coiled-coil domain containing 115, transcript variant X2
NC_018723.3_mrna_XM_003980243.4_8	XM_003980243.4	IMP4, U3 small nucleolar ribonucleoprotein
NC_018723.3_mrna_XM_023250065.1_9	XM_023250065.1	protein tyrosine phosphatase, non-receptor type 18
NC_018723.3_mrna_XM_023250181.1_10	XM_023250181.1	neurogenic locus notch homolog protein 3-like

Script’s output file → FILE1 with the gene product description on the last column

Long_geneID	logFC.2h	logFC.17h	logCPM	LR	PValue	FDR	Description
NC_018726.3_mrna_XM_023252992.1_17213	-0.0115539525378653	12.3552392903438	2.86329600280231	153.177804483085	5.46842652825283e-34	3.97729598252885e-29	heterogeneous nuclear ribonucleoprotein D, transcript variant X6
NC_018737.3_mrna_XM_023245341.1_62253	-2.41190133112974e-08	10.7952771226017	1.31660408910246	123.483120559081	1.53455008109868e-27	5.10785193903218e-23	CCCTC-binding factor, transcript variant X1
NC_018729.3_mrna_XM_023256742.1_28810	0.000496516357557754	10.763096713483	1.28333375331224	122.849205260556	2.10685197947211e-27	5.10785193903218e-23	peroxisomal biogenesis factor 5, transcript variant X1
NC_018736.3_mrna_XM_019818127.2_58617	-9.33346500212488	-10.032294222076	0.293441063845244	111.291091024287	6.81469842628745e-25	1.23911661485185e-20	phosphatidylinositol transfer protein, cytoplasmic 1, transcript variant X2
NC_018736.3_mrna_XM_019818041.1_58415	-8.34999208356182	-9.51603983117479	0.890988635810215	108.822101347149	2.34196789867635e-24	3.40672018413056e-20	integrin subunit beta 3, transcript variant X1
NC_018731.3_mrna_XM_019839937.2_40571	-9.53613707062572	-9.53619935953011	-0.197449535373358	105.301303647077	1.36179722364347e-23	1.65077059450061e-19	ring finger protein 168, transcript variant X2
NC_018735.3_mrna_XM_023242979.1_55286	1.31546722469771	9.94937447648528	0.58790776418027	100.232406950434	1.71715451761346e-22	1.78417260535803e-18	vav guanine nucleotide exchange factor 2, transcript variant X7
NC_018741.3_mrna_XM_023249335.1_70881	0.450036882025638	9.6533406059691	0.228666387292216	99.2434452989686	2.81552683373531e-22	2.55973622089046e-18	lysine demethylase 5C, transcript variant X1
NC_018724.3_mrna_XM_019822718.2_7002	3.71124136142327e-08	9.38866457539553	-0.0287171084086663	97.1391471277462	8.06310309653979e-22	6.51606238241702e-18	polybromo 1, transcript variant X23

→ The following is the same code from previous slide written in a single command line to be executed straight from the terminal prompt:

```
$ perl -e 'open(FILE1, "x.tsv"); open(FILE2, “y.tsv"); while(<FILE2>) {chomp($_); @array2 = split(/\t/, $_); $hash{$array2[0]} = $array2[1];} while(<FILE1>) {chomp($_); @array1 = split(/\t/, $_); if($hash{$array1[0]} ne "") {print("$_\t$hash{$array1[0]}\n");} else {print("$_\tNA\n");}}’ >z.tsv
```

Bring your issues on!