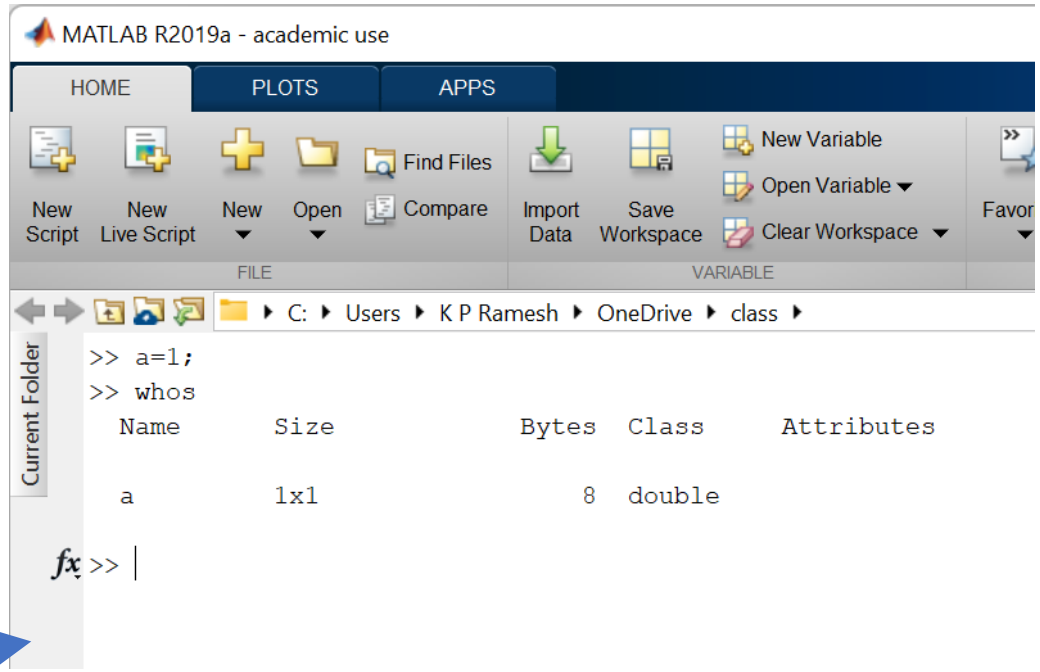
Data types and operators

Int, float, char, Arithmetic , relational and logical operators

MATLAB Data types

- Data type - what **type** of **value** a **variable** has and what type of mathematical, relational or logical **operations** can be **applied** to it.
- **Matrix** or **Array** is the **primary data type** in MATLAB.
- In MATLAB, all numeric variables are stored as **double-precision floating-point (64 bits)**.
- Type >> a=1;
- >> whos



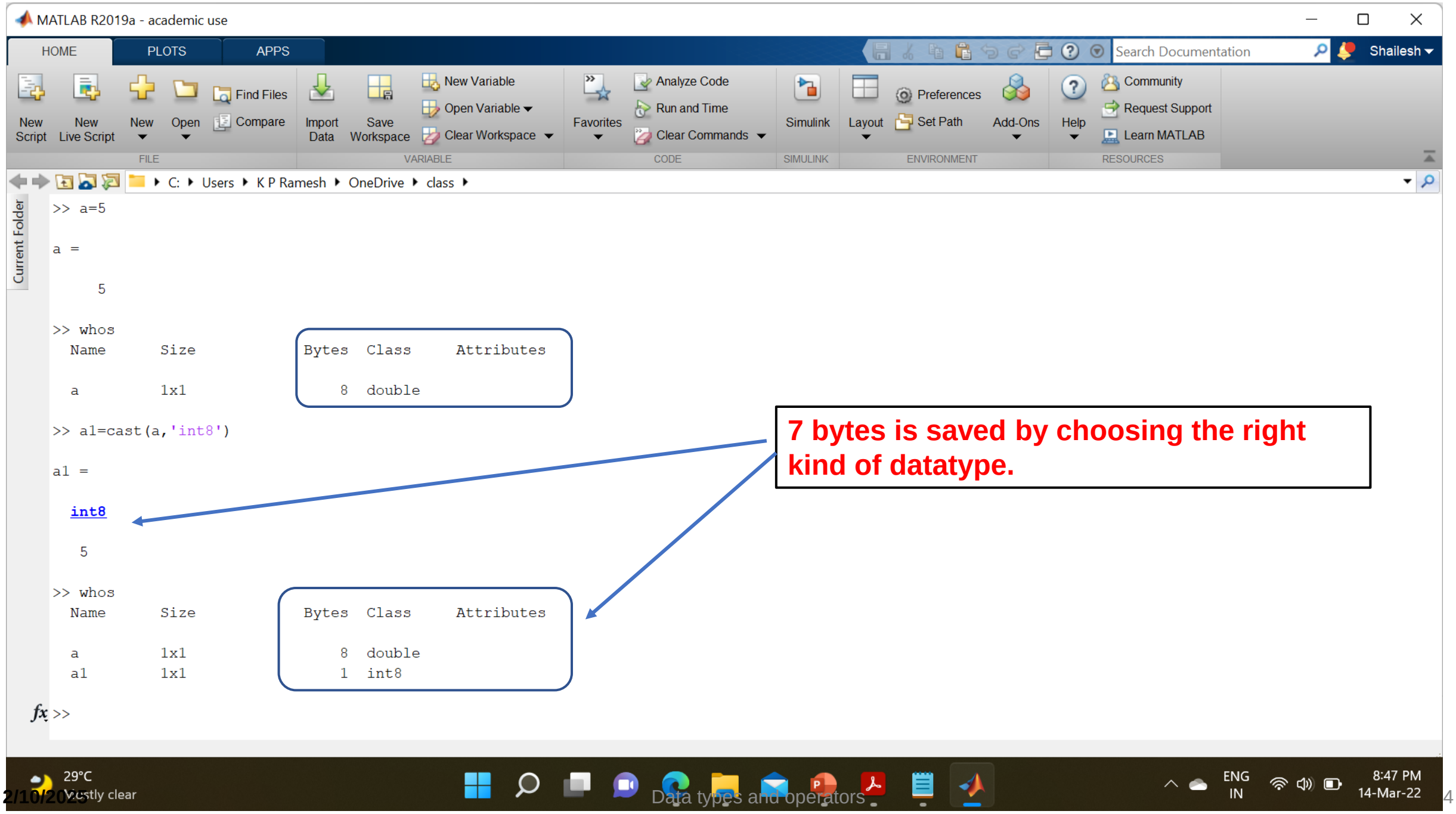
Numeric data types

- **int8** - 8-bit signed integer
- **uint8** - 8-bit unsigned integer
- **int16** - 16-bit signed integer
- **uint16** - 16-bit unsigned integer
- **int32** - 32-bit signed integer
- **uint32** - 32-bit unsigned integer
- **int64** - 64-bit signed integer
- **uint64** - 64-bit unsigned integer
- **single** - single precision numerical data
- **double** - double precision numerical data
- **logical** - 1 or 0

The basic strategy for selecting the best data type is to **select** the **smallest data type** that **matches** the **kind of data** you have and that allows for **all the feasible values of your data**.

```
type >>intmin('int8')
type >>intmax('int8')

type >>intmin('uint8')
type >>intmax('uint8')
```


MATLAB R2019a - academic use
HOME
PLOTS
APPS
Search Documentation
Shailesh
New Script
New Live Script
New
Open
Find Files
Compare
Import Data
Save Workspace
New Variable
Open Variable
Clear Workspace
Favorites
Analyze Code
Run and Time
Clear Commands
Simulink
Layout
Preferences
Set Path
Add-Ons
Help
Community
Request Support
Learn MATLAB
FILE
VARIABLE
CODE
SIMULINK
ENVIRONMENT
RESOURCES
C: ▸ Users ▸ K P Ramesh ▸ OneDrive ▸ class ▸
Current Folder
>> a=5
a =
5
>> whos
Name Size Bytes Class Attributes
a 1x1 8 double
>> a1=cast(a,'int8')
a1 =
int8
5
>> whos
Name Size Bytes Class Attributes
a 1x1 8 double
a1 1x1 1 int8
>>
7 bytes is saved by choosing the right kind of datatype.
29°C
Mostly clear
ENG
IN
8:47 PM
14-Mar-22

Characters and strings

- String scalar is created by enclosing a piece of text in double quotes.
- >> **Str=“Hello world!”;**
- >> **Str1=["Hello", "World"]** - string array, by concatenated string scalars using square brackets

- Character vector is created using single quotation marks.
- >> **C = 'Hello, world’**
- >> **B =char('MIT','Manipal’)**
- >> **C1 = char(Str);** - string scalar to char array

Concatenation / Joining strings

- >> **strcat**(‘MIT’, ‘Manipal’) - **'MITManipal’**
  - Concatenate strings horizontally

- >> str = ["Carlos","Sada";
- "Ella","Olsen";
- "Diana","Lee"]

SPLIT :
newStr = **split**(str) - divides str at whitespace character
newStr = **split**(str,delimiter) -divides each element of str at the delimiters

- >> newStr = **join**(str,"--");
  - combines the text in str and places the **elements of delimiter** between the elements of str instead of a space character.

Convert Between Numeric and Strings

- Convert numbers to character array
- >>s = **num2str**(pi);

- Convert character array or string to numeric array
- >> X = **str2num**('100 200 300 400’);

Find

- str = ["Mary Ann Jones","Paul Jay Burns","John Paul Smith"];
- pat = "Paul";
- TF = **contains**(str,pat); % returns a logical array

- str = "paired with red shoes";
- A = **count**(str,"red");

- str = 'Find the starting indices of substrings in a character vector';
- k = **strfind**(str,'in')

Replace

- newStr = **replace**(str,old,new) - Find and replace one or more substrings

- newStr = **strrep**(str,old,new) - Find and replace substrings

Extract

- Extract substrings from strings
- newStr = **extract**(str,**pat**) - returns any substrings in str that match the **pattern** specified by **pat**. (Example – digitsPattern, lettersPattern)

- newStr = **extract**(str,**pos**) - returns the character in str at the position specified by **pos**.

extractAfter	Extract substrings after specified positions
extractBefore	Extract substrings before specified positions
extractBetween	Extract substrings between start and end points

Dates and Time

- Arrays of date and time values that can be displayed in different formats.
- **clock** - Current date and time as date vector
- **cputime-** CPU time used by MATLAB expressed in seconds.
- **date** - Current date as character vector
- t = **datetime** - scalar datetime array corresponding to the current date and time.
- t = **datetime**('now','TimeZone','Asia/Seoul','Format','d-MMM-y HH:mm:ss Z')

MATLAB operators

Arithmetic Operators, Relational Operators, Logical Operators

Arithmetic operators

- Addition +
- Subtraction –
- Element-wise multiplication .*
- Matrix multiplication *
- Element-wise power .^

Symbol	Operation	MATLAB form
^	exponentiation: a^b	`a^b`
*	multiplication: ab	`a*b`
/	right division: $a/b = \frac{a}{b}$	`a/b`
\	left division: $a\backslash b = \frac{b}{a}$	`a\b`
+	addition: $a + b$	`a+b`
−	subtraction: $a - b$	`a-b`

^	Exponential Operator	6
*	Multiplication	7
/	Division	7
+	Addition	8
-	Subtraction	8

Order of precedence

Precedence	Operation
First	Parentheses, evaluated starting with the innermost pair.
Second	Exponentiation, evaluated from left to right.
Third	Multiplication and division with equal precedence, evaluated from left to right.
Fourth	Addition and subtraction with equal precedence, evaluated from left to right.

Relational and logical Operators

- == Equal to
- ~= Not equal to
- > Greater than
- >= Greater than or equal to
- < Less than
- <= Less than or equal to

Logical Operators

&&	logical AND
||	logical OR
~	logical NOT

expr1 && expr2
expr1 || expr2

Exercise

- What is the output of the following expressions? Give reasons.

1. >> 8 + 3*5
2. >> (8 + 3)*5
3. >> 4^2 - 12 - 8/4*2
4. >> 4^2 - 12 - 8/(4*2)
5. >> 3*4^2 + 5
6. >> (3*4)^2 + 5
7. >> 27^(1/3) + 32^(0.2)
8. >> 27^(1/3) + 32^0.2
9. >> 27^1/3 + 32^0.2

ANSWERs

- Output of the following expressions.

1. >> 8 + **3*5** = 8+**15** = **23**
2. >> **(8 + 3)***5 = **11** * 5 = 55
3. >> **4^2** - 12 - **8/4***2 = 16 – 12 – 2*2 = 0
4. >> **4^2** - 12 - 8/**(4*2)** = 16 – 12 – 8/8 = 3
5. >> 3***4^2** + 5 = 3*16+5 = 53
6. >> **(3*4)**^2 + 5 = 149
7. >> 27^**(1/3)** + 32^**(0.2)** = 27^**0.333** + 2 = 5
8. >> 27^(1/3) + 32^0.2 = 5
9. >> **27^1**/3 + 32^0.2 = 9+2 = 11