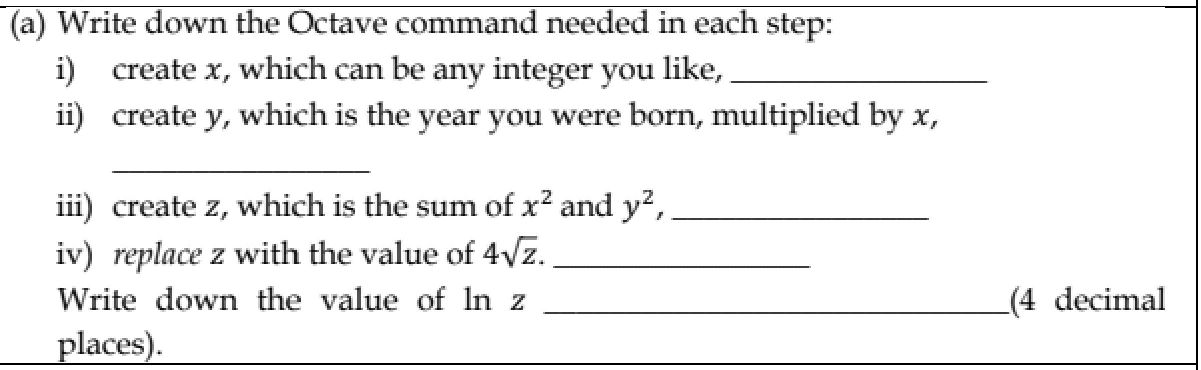
Name : IZZAR SULY NASHRUDIN | NIM : 230601110010

Algorithm & Programming: Octave Worksheets II

N	Info	Worksheets
1	Variabel s	



(a) Write down the Octave command needed in each step:

i) create x, which can be any integer you like, ________________

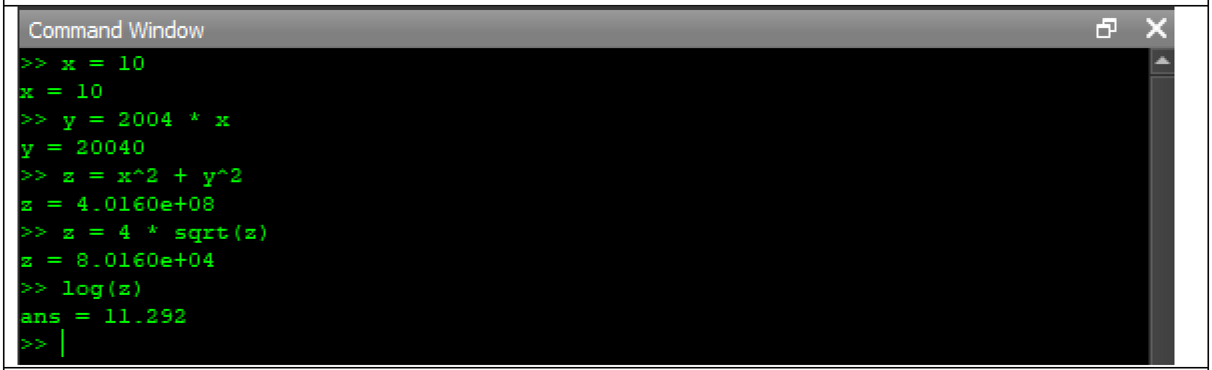
ii) create y, which is the year you were born, multiplied by x, ________________

iii) create z, which is the sum of x^2 and y^2, ________________

iv) *replace* z with the value of $4\sqrt{z}$. ________________

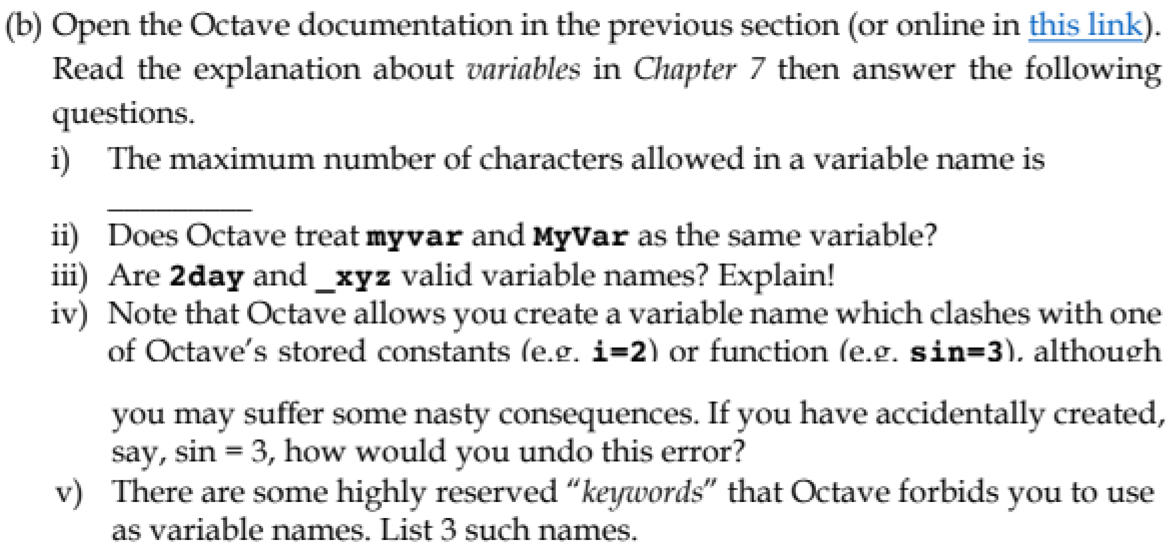
Write down the value of ln z ______________________________(4 decimal places).

Practice



```
>> x = 10
x = 10
>> y = 2004 * x
y = 20040
>> z = x^2 + y^2
z = 4.0160e+08
>> z = 4 * sqrt(z)
z = 8.0160e+04
>> log(z)
ans = 11.292
>> |
```

Worksheets



(b) Open the Octave documentation in the previous section (or online in this link). Read the explanation about *variables* in *Chapter 7* then answer the following questions.

i) The maximum number of characters allowed in a variable name is __________

ii) Does Octave treat **myvar** and **MyVar** as the same variable?

iii) Are **2day** and **_xyz** valid variable names? Explain!

iv) Note that Octave allows you create a variable name which clashes with one of Octave's stored constants (e.g. **i=2**) or function (e.g. **sin=3**), although you may suffer some nasty consequences. If you have accidentally created, say, sin = 3, how would you undo this error?

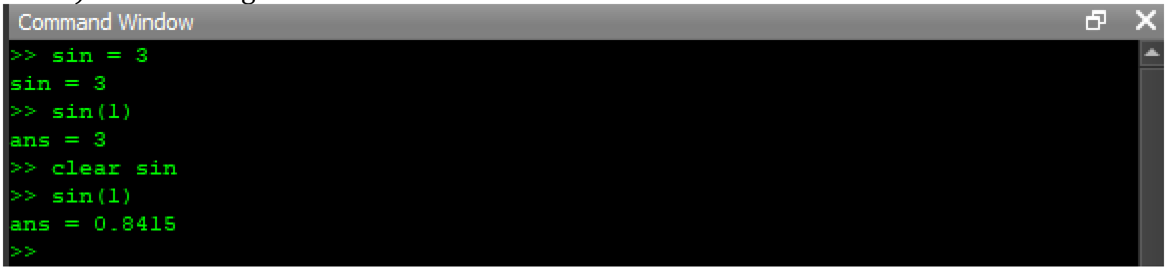
v) There are some highly reserved "*keywords*" that Octave forbids you to use as variable names. List 3 such names.

Answer

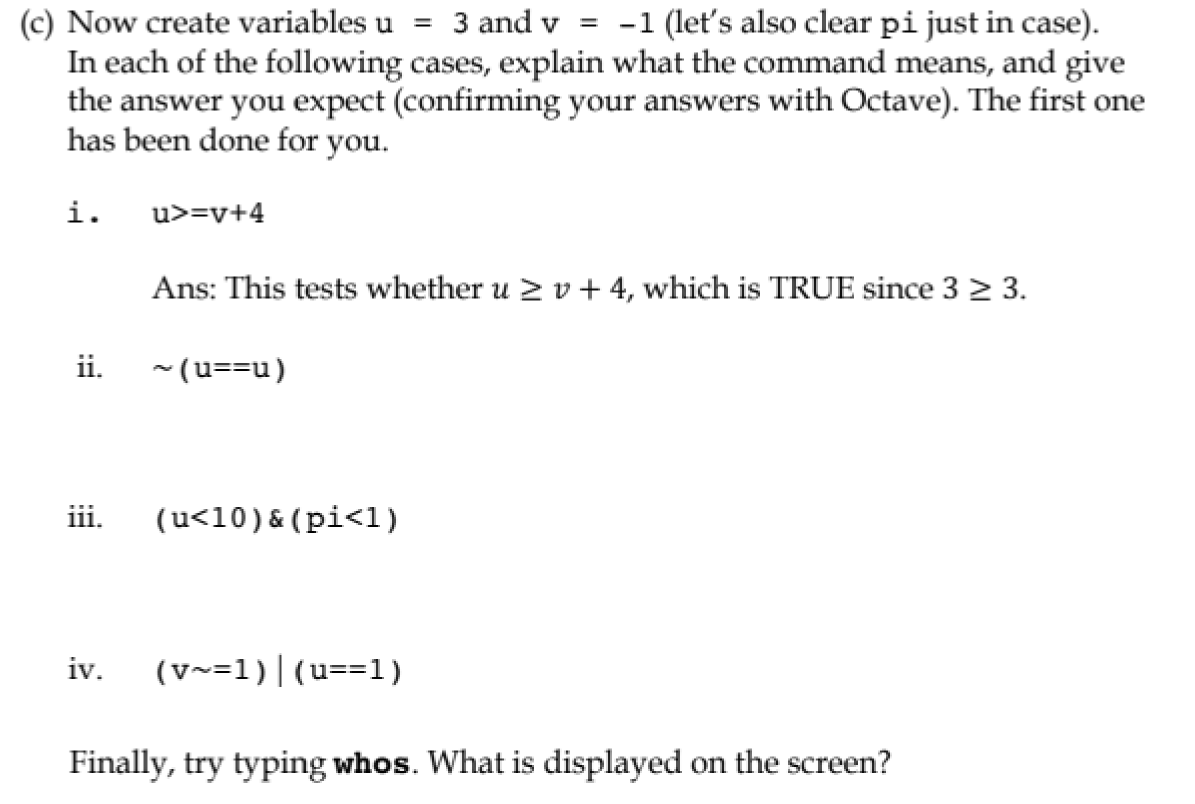
i) The name of a variable must be a sequence of letters, digits and underscores, but it may not begin with a digit. Octave does not enforce a limit on the length of variable names, but it is seldom useful to have variables with names longer than about **30 characters**

ii) Not, case is significant in variable names. The symbols **myvar** and **Myvar** are **distinct variables**. A variable name is a valid expression by itself. It represents the variable's current value. Variables are given new values with assignment operators and increment operators.

iii) For **2day** is an invalid variable because each variable name cannot begin with a number. names like **_xyz** that begin with underscores are understood to be reserved for internal use by Octave.

		iv) we can get the command **clear sin** to return the definition of the variable sin
		
		
		`>> sin = 3`
		`sin = 3`
		`>> sin(1)`
		`ans = 3`
		`>> clear sin`
		`>> sin(1)`
		`ans = 0.8415`
		`>>`
		v) 1. **i** and **j** : Typically used to represent the imaginary unit in complex numbers.
		2. Mathematical function names like **sin**, **cos**, **exp**, and others.
		3. Mathematical constants like **pi**, **e**, and **Inf**.

2	Types	**Worksheets** i) Computer memory is commonly measured is *bit* or *byte*. Explain what they mean. ii) How much memory is needed to store a variable of type **double**? ___________ We will explore many other types of variables as we go along. **Answer** i) **Bit (binary digit)**: A bit is the smallest unit of data in a computer and can have one of two values: 0 or 1. It represents the basic unit of information storage and processing in computing systems. Bits are often grouped together to represent larger units of data. **Byte**: A byte consists of 8 bits and is the fundamental unit of data storage and communication in computer systems. Bytes are used to represent characters, numbers, and other types of data. For example, a single ASCII character requires one byte (8 bits) of storage. Bytes are widely used to measure memory capacity and file sizes in computing. ii) A variable of type double typically requires 8 bytes of memory to store its value. This is true for many programming languages, including C, C++, Java, and others. So, Double (double-precision floating-point) variables are stored as IEEE 64-bit (8-byte) floating-point numbers ranging in value from: **-1.79769313486231E308** to **-4.94065645841247E-324** for negative values and **4.94065645841247E-324** to **1.79769313486232E308** for positive values
3	Logical	**Worksheets** Sometimes 1 and 0 are used to denote true or false, but which is which? Ans: ________________ Try typing the LHS of these statements into Octave (try the 0,1 notation also). (a) Complete the following truth tables for the AND and OR operators. & 1 0 1 0 | 1 0 1 0 (b) What is the difference between the following commands? `pi==3` and `pi=3`



(c) Now create variables `u = 3` and `v = -1` (let's also clear `pi` just in case). In each of the following cases, explain what the command means, and give the answer you expect (confirming your answers with Octave). The first one has been done for you.

i. `u>=v+4`

Ans: This tests whether $u \geq v + 4$, which is TRUE since $3 \geq 3$.

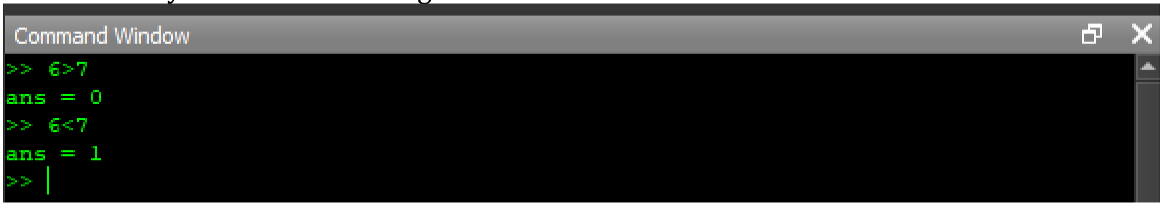
ii. `~(u==u)`

iii. `(u<10)&(pi<1)`

iv. `(v~=1)|(u==1)`

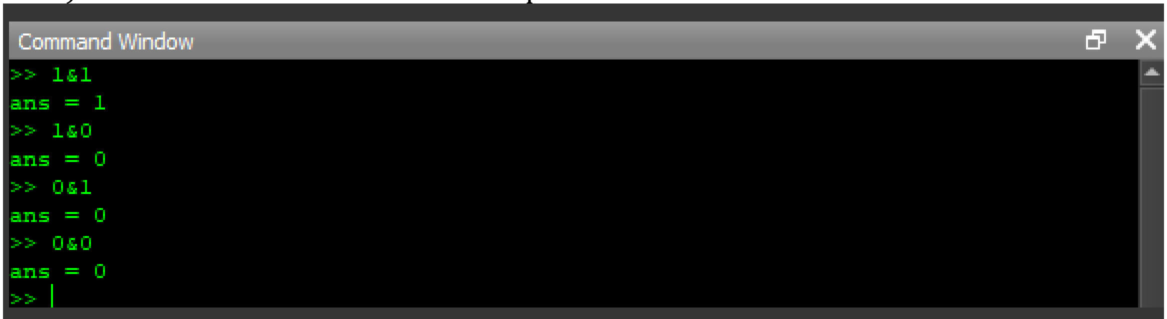
Finally, try typing **whos**. What is displayed on the screen?

Sometimes 1 and 0 are used to denote true or false, but which is which? In programming, **1 is often used to represent true, while 0 is used to represent false**. This is because these values are commonly used in Boolean logic.

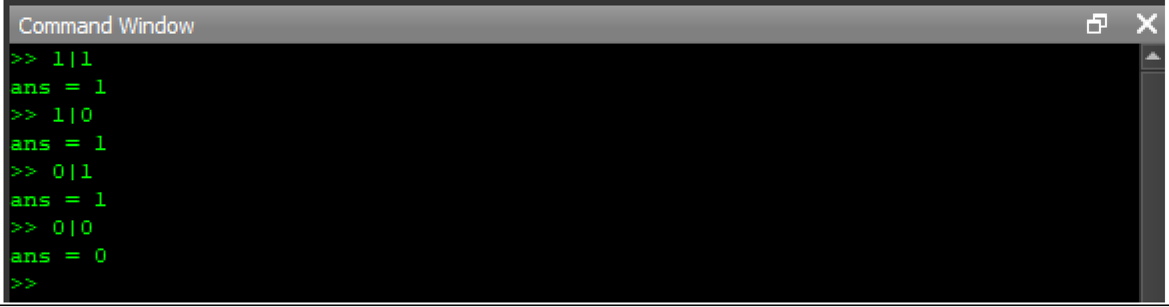


```
>> 6>7
ans = 0
>> 6<7
ans = 1
>> 
```

a) Truth tables for the AND and OR operators.



```
>> 1&1
ans = 1
>> 1&0
ans = 0
>> 0&1
ans = 0
>> 0&0
ans = 0
>> 
```

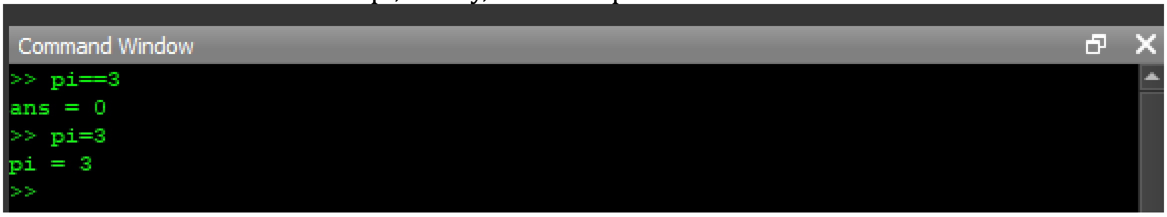


```
>> 1|1
ans = 1
>> 1|0
ans = 1
>> 0|1
ans = 1
>> 0|0
ans = 0
>>
```

&	1	0
1	1	0
0	0	0

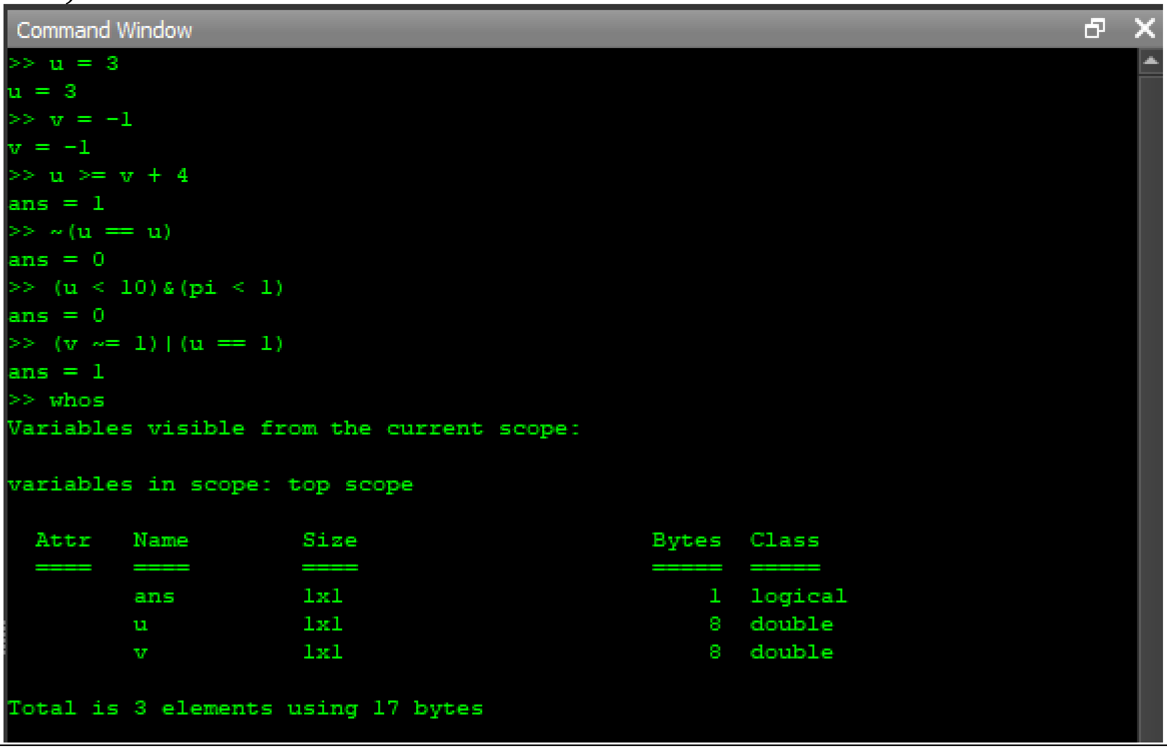
|	1	0
1	1	1
0	1	0

b) "**pi == 3**" is a comparison or logical expression that checks if the value of the variable "**pi**" is equal to 3. If "pi" has a value equal to 3, then the expression evaluates to true; otherwise, it evaluates to false. On the other hand, "**pi = 3**" is an assignment statement where the value 3 is assigned to the variable "**pi**". With this statement, the previous value of the variable "pi," if any, will be replaced with the value 3.



```
>> pi==3
ans = 0
>> pi=3
pi = 3
>>
```

c) Practice



```
>> u = 3
u = 3
>> v = -1
v = -1
>> u >= v + 4
ans = 1
>> ~(u == u)
ans = 0
>> (u < 10)&(pi < 1)
ans = 0
>> (v ~= 1)|(u == 1)
ans = 1
>> whos
Variables visible from the current scope:

variables in scope: top scope

  Attr   Name        Size                     Bytes  Class
  ====   ====        ====                     =====  =====
         ans         1x1                          1  logical
         u           1x1                          8  double
         v           1x1                          8  double

Total is 3 elements using 17 bytes
```