Exercise

- Write a program called `age.py` that asks the user to enter his/her age and based on it, prints the following message/s
  - If a person is at least 16, the program prints "You are eligible to drive"
  - If a person is at least 18, the program prints "You are eligible to drive" and "And you can also vote or join the military"
  - If a person is at least 21, the program also prints "You are eligible to drive" and "And you can also vote or join the military" and "Guess what – you can drink alcohol"
  - If a person is less than 16, the program prints "Stay in school and enjoy what's remaining of your childhood"

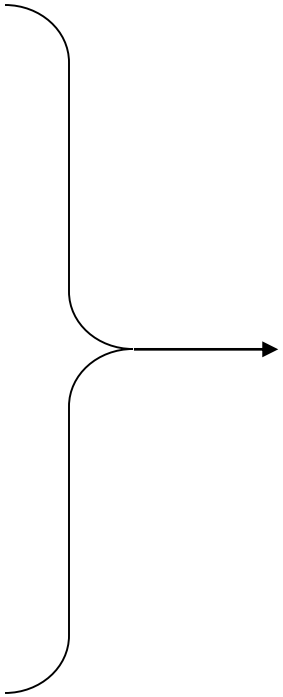
Test your program with the following values: 14, 16, 18 and 21

Factoring `if/else` code

- **factoring**: Extracting common/redundant code.
  - Can reduce or eliminate redundancy from `if/else` code.

- Example:

```
if a == 1:
    print(a)
    x = 3
    b = b + x
elif a == 2:
    print(a)
    x = 6;
    y = y + 10;
    b = b + x
elif a == 3:
    println(a)
    x = 9
    b = b + x
```



```
print(a)
x = 3 * a
if a == 2:
    y = y + 10
b = b + x
```

Comparing Floats

- Most decimal fractions cannot be represented as binary fractions – decimal floating-point numbers are only approximated by the binary ones
  - enter `0.1 + 0.1 + 0.1`

- Floating point numbers should not be compared for equality but rather for near equality

  instead of
  ```
  if a == 4.0
  ```
  ```
  if abs(a - 4.0) < 0.00000001
  ```

Comparing Strings

- String comparison is based on character by character comparison of string contents.

- How are characters compared and what determines which character is considered smaller / larger?
  - Lookup tables

Character Lookup Tables

- Bitstreams are matched to their character equivalent through look-up tables for the specific character set
  - This is an ASCII table

- Looking up
  - Example: 1001000

	xxxyyyy	xxx							
		000	001	010	011	100	101	110	111
	0000	NUL	DLE	space	0	@	P	`	p
	0001	SOH	DC1	!	1	A	Q	a	q
	0010	STX	DC2	"	2	B	R	b	r
	0011	ETX	DC3	#	3	C	S	c	s
	0100	EOT	DC4	$	4	D	T	d	t
	0101	ENQ	NAK	%	5	E	U	e	u
y	0110	ACK	SYN	&	6	F	V	f	v
y	0111	BEL	ETB	'	7	G	W	g	w
y	1000	BS	CAN	(	8	H	X	h	x
y	1001	TAB	EM	)	9	I	Y	i	y
	1010	LF	SUB	*	:	J	Z	j	z
	1011	VT	ESQ	+	;	K	[	k	{
	1100	FF	FS	,	<	L	\	l	|
	1101	CR	GS	-	=	M	]	m	}
	1110	SO	RS	.	>	N	^	n	~
	1111	SI	US	/	?	O	_	o	DEL

ord and chr

- Function `ord` takes a character as an argument and returns its numeric equivalent, e.g.

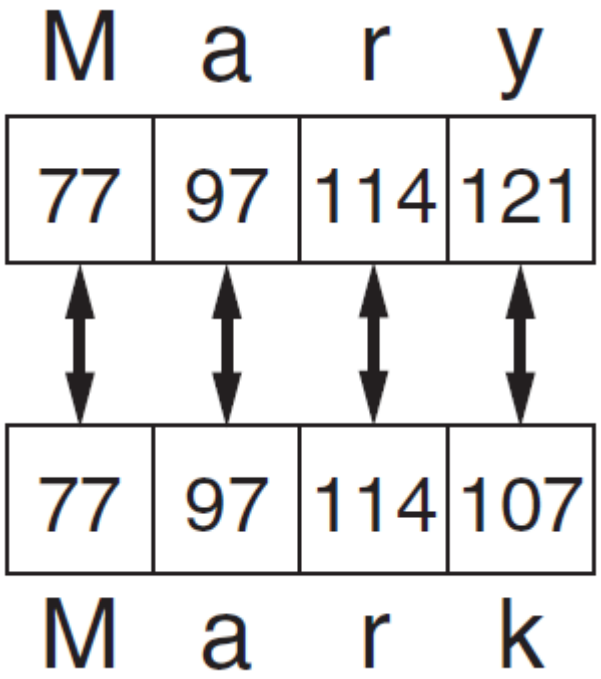
```
>>> ord('a')
```

- Function `chr` takes a number as an argument and returns its character equivalent, e.g.

```
>>> chr(100)
```

String Comparison

- Strings are compared character by character based on the ASCII values for each character



  – If shorter word is substring of longer word, longer word is greater than shorter word