pythex

Link to this regex

Your regular expression:

/ (\b(?:be|am|is|are|was|were|have|has|had)\b[\w\s]{,15}?(?:d|(?<!whe)n|n|e|left|t|being)\b(?: by\b)?) /

IGNORECASE | MULTILINE | DOTALL | VERBOSE

Your test string:

Not all of these are written in passive voice.
The seashell was found by the girl in the white hat.
The vegetables were disliked by the children.
The movie was enjoyed by all.
The decorations for the party were created by Jessica.
The phone was left in the car.

Match result:

Not all of these are written in passive voice.
The seashell was found by the girl in the white hat.
The vegetables were disliked by the children.
The movie was enjoyed by all.
The decorations for the party were created by Jessica.
The phone was left in the car.
The vase was broken. (Passive voice is appropriate when you don't know who performed the action).
The toy was chewed by the dog.
The stuffed animal was well loved by the little girl.
The laundry is always done by Mom.
These cookies were baked at the grocery store bakery.
A sentence is written in active voice when the subject of the sentence performs the action in the sentence.
A sentence is written in passive voice when the subject of the sentence has an action done to it by someone or something else.
A stone smashed the window.
Daniel was watching the birds.
The dog was being washed by the girl.
The girl was washing the dog.
Fred told Johnny a lie.
James climbed the ladder.
James couldn’t see the man.
James hit the tree with his stick.
Mark was eating an apple.
Mark was given a warning.
Sam baked a big cake.
Susan found her car keys.
The actors had performed the play by Shakespeare.
The boy picked up the coin.
The boys pushed the tree over.
The car had been driven into a wall by a naughty child.
The car was fixed.
The card was made by Fred.

Match captures:

Match 1
1. are written

Match 2
1. was found by

Match 3
1. were disliked by

Match 4
1. was enjoyed by

Match 5
1. were created by

Match 6
1. was left

Match 7
1. was broken

Match 8
1. was chewed by

Match 9
1. was well loved by

Match 10

The crisp packet was thrown away.
The egg was laid by the bird.
The football was kicked by Luke.
The key was used to open the door.
The knife was left on the table by Julie.
The man jumped off the step.
The milk had been knocked over by a cat.
The pencil had been lost.
The phone was being used by Mr Thomas.
The picture was painted by Bob.
The policeman chased after Fred.
The windows had been washed.
Tina opened the present.

1. `is always done by`

Match 11

1. `were baked`

Match 12

1. `is written`

Match 13

1. `is written`

Match 14

1. `has an`

Match 15

1. `was being`

Match 16

1. `was eating an`

Match 17

1. `was given`

Match 18

1. `had performed`

Match 19

1. `had been`

Match 20

1. `was fixed`

Match 21

1. `was made by Fred`

Match 22

1. `was thrown`

Match 23

1. `was laid by`

Match 24

1. `was kicked by`

Match 25

1. `was used`

Match 26

1. `was left`

Match 27

1. `had been`

Match 28

1. `had been`

Match 29

1. `was being`

Match 30

1. `was painted by`

Match 31

1. `had been`

Regular expression cheatsheet

Special characters

`\`	escape special characters
`.`	matches any character
`^`	matches beginning of string
`$`	matches end of string
`[5b-d]`	matches any chars '5', 'b', 'c' or 'd'
`[^a-c6]`	matches any char except 'a', 'b', 'c' or '6'
`R|S`	matches either regex `R` or regex `S`
`()`	creates a capture group and indicates precedence

Quantifiers

`*`	0 or more (append `?` for non-greedy)
`+`	1 or more (append `?` for non-greedy)
`?`	0 or 1 (append `?` for non-greedy)
`{m}`	exactly `m` m occurrences
`{m, n}`	from `m` to `n`. `m` defaults to 0, `n` to infinity
`{m, n}?`	from `m` to `n`, as few as possible

Special sequences

`\A`	start of string
`\b`	matches empty string at word boundary (between `\w` and `\W`)
`\B`	matches empty string not at word boundary
`\d`	digit
`\D`	non-digit
`\s`	whitespace: `[ \t\n\r\f\v]`
`\S`	non-whitespace
`\w`	alphanumeric: `[0-9a-zA-Z_]`
`\W`	non-alphanumeric
`\Z`	end of string
`\g<id>`	matches a previously defined group

Special sequences

`(?iLmsux)`	matches empty string, sets re.X flags
`(?:...)`	non-capturing version of regular parentheses
`(?P...)`	matches whatever matched previously named group
`(?P=)`	digit
`(?#...)`	a comment; ignored
`(?=...)`	lookahead assertion: matches without consuming
`(?!...)`	negative lookahead assertion
`(?<=...)`	lookbehind assertion: matches if preceded
`(?<!...)`	negative lookbehind assertion
`(?(id)yes|no)`	match 'yes' if group 'id' matched, else 'no'

Based on tartley's python-regex-cheatsheet.

Inspired by Rubular. For a complete reference, see the official `re` module documentation.

Made by Gabriel Rodríguez. Powered by Flask and jQuery.