Improving Interpretability and Understandability of NumPy Program Synthesis

Kevin Ye and Jacob Yim - in collaboration with Justin Lubin and Prof. Sarah Chasins

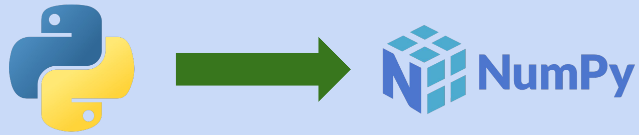
EPIC DATA lab
UC Berkeley

NumPy program synthesis

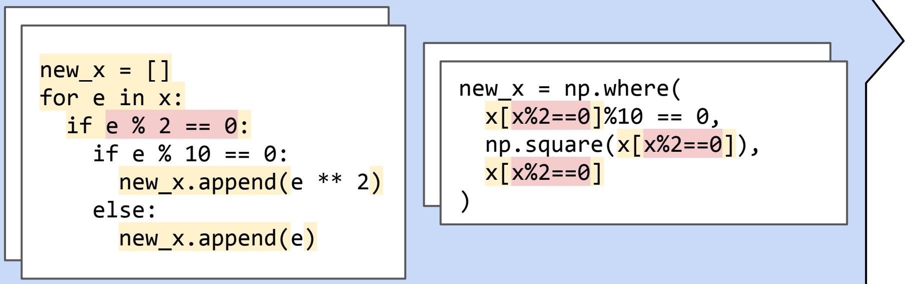
Easier to write for novice programmers

More difficult to write, but better performance on large datasets

Using component-based refactoring:



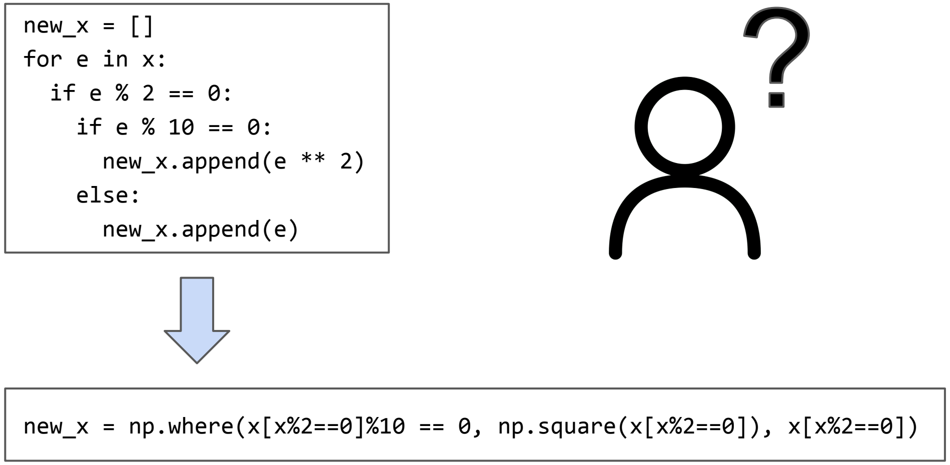
Many possible highlighting schemes!



```
new_x = []
for e in x:
  if e % 2 == 0:
    if e % 10 == 0:
      new_x.append(e ** 2)
    else:
      new_x.append(e)
```

```
new_x = np.where(
  x[x%2==0]%10 == 0,
  np.square(x[x%2==0]),
  x[x%2==0]
)
```

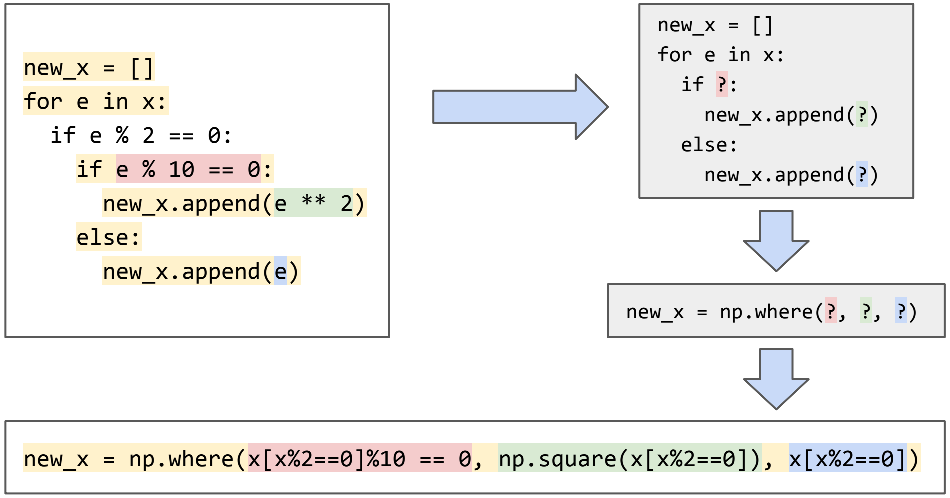
Existing synthesizer



```
new_x = []
for e in x:
  if e % 2 == 0:
    if e % 10 == 0:
      new_x.append(e ** 2)
    else:
      new_x.append(e)
```

```
new_x = np.where(x[x%2==0]%10 == 0, np.square(x[x%2==0]), x[x%2==0])
```

New user interfaces



```
new_x = []
for e in x:
  if e % 2 == 0:
    if e % 10 == 0:
      new_x.append(e ** 2)
    else:
      new_x.append(e)
```

```
new_x = []
for e in x:
  if ?:
    new_x.append(?)
  else:
    new_x.append(?)
```

```
new_x = np.where(?, ?, ?)
```

```
new_x = np.where(x[x%2==0]%10 == 0, np.square(x[x%2==0]), x[x%2==0])
```