Code generator

Code generator takes input file in json format and produce a “output.cpp” file in desired format according to template which is provided.

Solution description:

The base idea is that the code saves content of template.cpp to a variable “template” as a string. Then prepares two string variables (“items_to_insert” and ”colNames_to_insert”) which are formatted in desired way and contain data extracted from input file. Once those variables are prepared, the code inserts these two variables to “template” variable in a right place. This is ensured by retrieving indexes of two key words (//items and //collections) that are included in template.cpp. Updated “template” variable is then written to new “output.cpp” file.

Steps:

1.Import input.json file and stores its content to data variable using json library.

```
# Get the file name from the command line
jsonFile = sys.argv[1]

# Open the file and read the data
with open(jsonFile, 'r') as file:
  data = json.load(file)
```

2.Using nested for loop to iterate through “data” to extract item values of each collection attribute. Items values are then saved as string in desired format to “items_to_insert” variable. Outcome of “items_to_insert” variable is string containing formatted item values for each attribute.

```
# Generate the code to insert
items_to_insert = ""
colNames_to_insert = ""
for coll_name in data.keys():
    for item in data[coll_name]:
        name, tag, type, size = item['name'], item['tag'], item['type'], item['size']
        items_to_insert += (f"initDataItem(\"{name}\",  {tag},  \"{type}\",  {size});\n    ")
        colNames_to_insert += (f"Colection.{coll_name}->push(\"{name}\");\n    ")
```

3. Opens template.cpp and saves its content to template variable

```
# Open the template file
with open('template.cpp', 'r') as template_file:
    template = template_file.read()
```

4. Searches for index of “//items” in template variable. Template variable is then updated with the content of items_to_insert variable. Since strings are immutable in python, new “template” variable is created by contatenating template’s content before index + items_to_insert + template after index (excluding fetched word “//items”)

```
#Inserting items
index_items = template.index("//items") # Get the index where we want to insert items
template = template[:index_items] + items_to_insert + template[index_items+7:]
```

5. Searches for index of “//collections” in template variable. Template variable is then updated with the content of colNames_to_insert variable. Since strings are immutable in python, new “template” variable is created by concatenating template’s content before index + colNames_to_insert + template after index (excluding fetched word “//collections”)

6. new file output.cpp is created and content of “template” variable is inserted.

```
# Open the output file
output_file = open('output.cpp', 'w')

# Write the template to the output file
output_file.write(template)
# Close the output file
output_file.close()
```