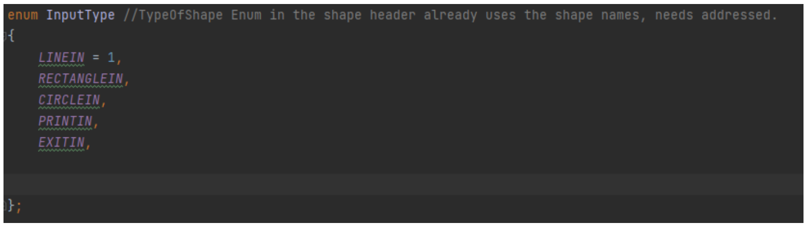
William Dragstrem
Professor Josiah Greenwell
Project Check-In 2
Shapes Project V1.2.0

Project Link:

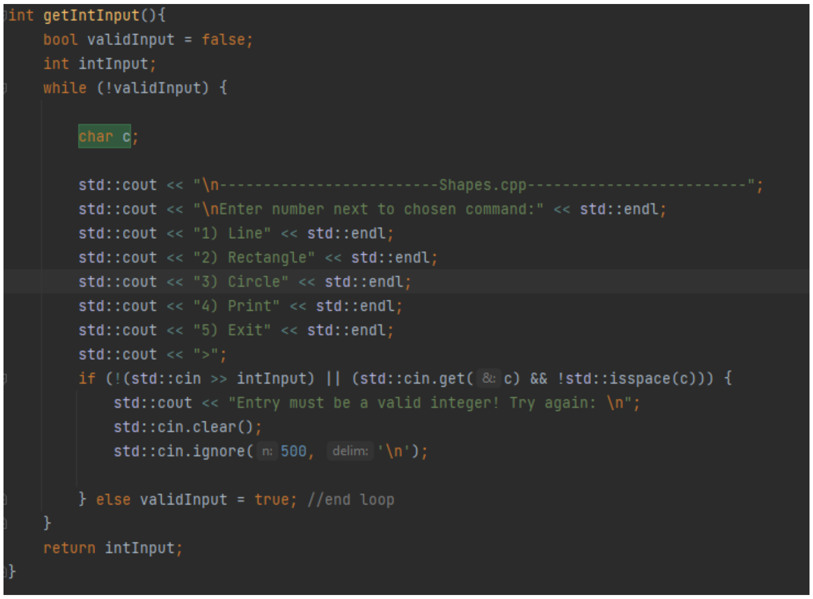
Information:
The original project can be found in the “main” branch. The most recent version, as of this document, is in the V-1.2.0 branch. All code updates listed occurred in ShapesProject.cpp.

Updates:

1.) Code has been reformatted so that the call for input to determine user selection is now encapsulated within a function. That function ensures that the user inputs only INTEGERS, as all possible inputs are now designated an enum so as to future proof for any additional shapes that could be included into this program that start with the same letter as another shape/input.



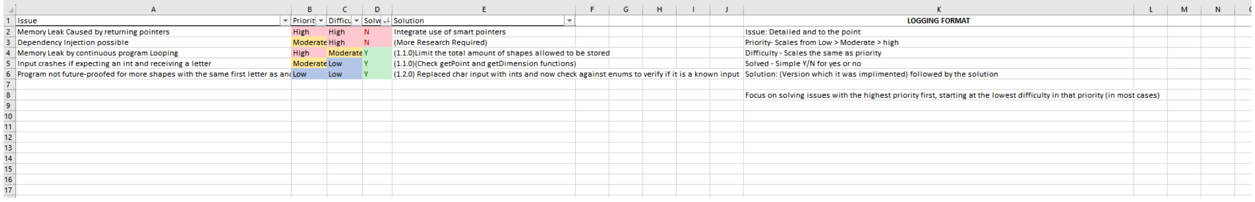
```
enum InputType //TypeOfShape Enum in the shape header already uses the shape names, needs addressed.
{
    LINEIN = 1,
    RECTANGLEIN,
    CIRCLEIN,
    PRINTIN,
    EXITIN,


};
```



```
int getIntInput(){
    bool validInput = false;
    int intInput;
    while (!validInput) {

        char c;

        std::cout << "\n------------------------Shapes.cpp------------------------";
        std::cout << "\nEnter number next to chosen command:" << std::endl;
        std::cout << "1) Line" << std::endl;
        std::cout << "2) Rectangle" << std::endl;
        std::cout << "3) Circle" << std::endl;
        std::cout << "4) Print" << std::endl;
        std::cout << "5) Exit" << std::endl;
        std::cout << ">";
        if (!(std::cin >> intInput) || (std::cin.get( &: c) && !std::isspace(c))) {
            std::cout << "Entry must be a valid integer! Try again: \n";
            std::cin.clear();
            std::cin.ignore( n: 500,  delim: '\n');

        } else validInput = true; //end loop
    }
    return intInput;
}
```

2.) A terms of use has now been written and added on to the README in the github repository

3.) The format for logging all known issues and changes to code has been created and standardized in excel.



	A	B	C	D	E	K
1	Issue	Priorit	Difficu	Solve	Solution	**LOGGING FORMAT**
2	Memory Leak Caused by returning pointers	High	High	N	Integrate use of smart pointers	Issue: Detailed and to the point
3	Dependency Injection possible	Moderate	High	N	(More Research Required)	Priority- Scales from Low > Moderate > high
4	Memory Leak by continuous program Looping	High	Moderate	Y	(1.1.0)Limit the total amount of shapes allowed to be stored	Difficulty - Scales the same as priority
5	Input crashes if expecting an int and receiving a letter	Moderate	Low	Y	(1.1.0)(Check getPoint and getDimension functions)	Solved - Simple Y/N for yes or no
6	Program not future-proofed for more shapes with the same first letter as anc	Low	Low	Y	(1.2.0) Replaced char input with ints and now check against enums to verify if it is a known input	Solution: (Version which it was implimented) followed by the solution
7						
8						Focus on solving issues with the highest priority first, starting at the lowest difficulty in that priority (in most cases)
9						
10						
11						
12						
13						
14						
15						
16						
17						

Future Updates:

1.) Prevent dependency injection

2.) Replace pointers with smart pointers