In Class Assignments: Week 7

1.Write an object-oriented C++ program to create a class for room and three objects to calculate the area and volume of three rooms. Use public access specifier and methods for calculations.

#include <iostream>  
using namespace std;  
class room{  
public:  
 float length, width, height;  
 void set\_var(float LENGTH, float WIDTH, float HEIGHT){  
 length = LENGTH;  
 width = WIDTH;  
 height = HEIGHT;  
 }  
 //Object for calculating Volume  
 //not passing parameter  
 float vol\_noPass(){  
 return length \* width \* height;  
 }  
 //Object for calculating Area  
 float area\_noPass(){  
 return length \* width;  
 }  
};  
  
int main() {  
 float l, w, h, volume, area;  
 room c{};  
  
 cout << "Enter Length, Width, Height Values: " << endl;  
 //set variables  
 cin >> l >> w >> h;  
 c.set\_var(l, w,h);  
  
 volume = c.vol\_noPass();  
 area = c.area\_noPass();  
 cout << "No-Passing Parameter Volume: " << volume << " units" << "\nNo-Passing Parameter Area: " << area << " units"<< endl;  
 return 0;  
}

Enter Length, Width, Height Values:

3

6

9

No-Passing Parameter Volume: 162 units

No-Passing Parameter Area: 162 units

Process finished with exit code 0

2. Write a program to find the volume of a rectangle using constructor (with and without passing parameters).

#include <iostream>  
using namespace std;  
  
class room{  
private:  
 float length, width, height;  
public:  
 /\*void set\_var(float LENGTH, float WIDTH, float HEIGHT){  
 length = LENGTH;  
 width = WIDTH;  
 height = HEIGHT;  
 }\*/  
 //not passing parameter  
 float vol\_noPass(){  
 cout << "Enter Length, Width, Height Values: " << endl;  
 cin >> length >> width >> height;  
 return length \* width \* height;  
 }  
 //passing parameter  
 void vol\_pass(float length,float width,float height){  
 cout << "Passing Parameter Volume: " << length \* width \* height << " units" << endl;  
 }  
};  
  
int main() {  
 float l{5}, w{10}, h{3}, volume;  
 room c{};  
  
 /\*//set length, width, height variables from main  
 c.set\_var(l, w,h);\*/  
  
 //find volume with passing a parameter  
 c.vol\_pass(5, 10, 3);  
 //find volume without passing a parameter  
 volume = c.vol\_noPass();  
 cout << "No-Passing Parameter Volume: " << volume << " units" << endl;  
 return 0;  
}

Passing Parameter Volume: 150 units

Enter Length, Width, Height Values:

5

10

3

No-Passing Parameter Volume: 150 units

Process finished with exit code 0

3. Write a C++ Program to show counter using Constructor.

#include <iostream>  
using namespace std;  
class counter{  
public:  
 int count{0}, count\_incr{1};  
 int get\_counterIncr(){  
 int incr;  
 cout << "Enter counter increment: [ ]\b\b";  
 cin >> incr;  
 count\_incr = incr;  
 }  
 void counter\_up(){  
 count += count\_incr;  
 }  
 void counter\_down(){  
 count -= count\_incr;  
 }  
 int disp\_counter(){  
 cout << "Count [" << count << "]" << endl;  
 }  
};  
  
int main() {  
 int usr\_choice;  
 counter c{};  
 c.get\_counterIncr();  
 cout << "\nEnter Choice\n[1]Count Up\n[2]Count Down\n" << endl;  
 cin >> usr\_choice;  
  
 if(usr\_choice == 1){  
 c.counter\_up();  
 }  
 else if(usr\_choice == 2){  
 c.counter\_down();  
 }  
 else{  
 cout << "Error: Invalid Input" << endl;  
 }  
 c.disp\_counter();  
 return 0;  
}

Enter counter increment: [7]

Enter Choice

[1]Count Up

[2]Count Down

1

Count [7]

4. Use Stack operations to

1. Display the elements in stack
2. get the size of stack
3. Print top of the stack
4. Pop some stack values and display the stack.

#include <iostream>  
#include <stack>  
using namespace std;  
  
//stores stack to be able to edit it  
class stackoperations{  
public:  
 stack<string> stack1;  
 string stack\_elem;  
};  
//derived class for UI funtions  
//not helpful, just throwing it in  
class ui\_elem: public stackoperations{  
public:  
 //Adds enough blank lines to make it seem as if the console was cleared.  
 //Every other built-in function I know of or could find was OS-dependant.  
 void clear\_console(){  
 cout << string(10, '\n') << endl;  
 }  
 //visually separates the output from the following menu prompt.  
 void in\_out\_sep(){  
 cout << string(100, '#') << "\n" << endl;  
 }  
 //Wraps user's menu selection with brackets.  
 void input\_box(){  
 cout << "[ ]\b\b";  
 }  
};  
//to access objects within class  
stackoperations s{};  
ui\_elem ui{};  
//(a)Display the elements in stack  
void displ\_menu(){  
 ui.in\_out\_sep();  
 cout << "Enter\n[1]Add Element\n[2]Remove Previous Element\n[3]Display Quantity of Elements\n[4]Call previously added Element\n"  
 "[5]Display All Elements\n[0]Exit" << endl;  
}  
//(1/2(d))Pop some stack values.  
void add\_elem(){  
 cin >> s.stack\_elem;  
 s.stack1.push(s.stack\_elem);  
 ui.clear\_console();  
}  
void rem\_elem(){  
 s.stack1.pop();  
 ui.clear\_console();  
}  
//(b)get the size of stack  
void displ\_stack\_size(){  
 //clearing before so that user can see ouput  
 //could use a continue-verification or timer  
 ui.clear\_console();  
 cout << "Quantity: " << s.stack1.size() << "\n" << endl;  
}  
//(c)Print top of the stack  
void displ\_last\_elem(){  
 ui.clear\_console();  
 cout << s.stack1.top() << "\n" << endl;  
}  
//(a)&(1/2(d))Display the elements in stack  
void displ\_all\_elem(){  
 ui.clear\_console();  
 //creating a copy of stack to use in function so that the original stack does not get modified.  
 stack<string> copy\_stack1 = s.stack1;  
 while(!copy\_stack1.empty()){  
 cout << copy\_stack1.top() << "\n";  
 copy\_stack1.pop();  
 }  
}  
  
int main(){  
 int usr\_choice;  
 do{  
 //Outputs menu options  
 displ\_menu();  
 ui.input\_box();  
 cin >> usr\_choice;  
 switch(usr\_choice){  
 case 0:  
 break;  
 case 1:  
 add\_elem();  
 break;  
 case 2:  
 rem\_elem();  
 break;  
 case 3:  
 displ\_stack\_size();  
 break;  
 case 4:  
 displ\_last\_elem();  
 break;  
 case 5:  
 displ\_all\_elem();  
 break;  
 }  
 }while(usr\_choice != 0);  
 return 0;  
}

####################################################################################################

Enter

[1]Add Element

[2]Remove Previous Element

[3]Display Quantity of Elements

[4]Call previously added Element

[5]Display All Elements

[0]Exit

[1]

I

####################################################################################################

Enter

[1]Add Element

[2]Remove Previous Element

[3]Display Quantity of Elements

[4]Call previously added Element

[5]Display All Elements

[0]Exit

[1]

H

####################################################################################################

Enter

[1]Add Element

[2]Remove Previous Element

[3]Display Quantity of Elements

[4]Call previously added Element

[5]Display All Elements

[0]Exit

[1]

---

####################################################################################################

Enter

[1]Add Element

[2]Remove Previous Element

[3]Display Quantity of Elements

[4]Call previously added Element

[5]Display All Elements

[0]Exit

[1]

H

####################################################################################################

Enter

[1]Add Element

[2]Remove Previous Element

[3]Display Quantity of Elements

[4]Call previously added Element

[5]Display All Elements

[0]Exit

[1]

I

####################################################################################################

Enter

[1]Add Element

[2]Remove Previous Element

[3]Display Quantity of Elements

[4]Call previously added Element

[5]Display All Elements

[0]Exit

[5]

I

H

---

H

I

####################################################################################################

Enter

[1]Add Element

[2]Remove Previous Element

[3]Display Quantity of Elements

[4]Call previously added Element

[5]Display All Elements

[0]Exit

[2]

####################################################################################################

Enter

[1]Add Element

[2]Remove Previous Element

[3]Display Quantity of Elements

[4]Call previously added Element

[5]Display All Elements

[0]Exit

[3]

Quantity: 4

####################################################################################################

Enter

[1]Add Element

[2]Remove Previous Element

[3]Display Quantity of Elements

[4]Call previously added Element

[5]Display All Elements

[0]Exit

[5]

H

---

H

I

####################################################################################################

Enter

[1]Add Element

[2]Remove Previous Element

[3]Display Quantity of Elements

[4]Call previously added Element

[5]Display All Elements

[0]Exit

[0]

Process finished with exit code 0