

## Creative Software Programming, Assignment 9-2

Handed out: Nov 11, 2021

**Due: 23:59, Nov 16, 2021 (NO SCORE for late submissions!)**

- Only files submitted by **git push to this course project at <https://hconnect.hanyang.ac.kr> (<Year>\_<Course no.>\_<Class code>/<Year>\_<Course no.>\_<Student ID>.git)** will be scored.
- Place your files under the directory structure <**Assignment name**>/<**Problem no.**>/<**your files**> just like the following example.
  - **DO NOT push CMake intermediate output files** (*CMakeCache.txt*, *cmake\_install.cmake*, *Makefile*, *CMakeFiles/\**).
  - **DO NOT use *cmake\_minimum\_required()* command in *CMakeLists.txt*.**

```
+ 2020_ITE0000_201900001
  + 2-1/
    + 1/
      - 1.cpp
      - CMakeLists.txt
    + 2/
      - 2.cpp
      - CMakeLists.txt
    + ...
```

- The submission time is determined not when the commit is made **but when the git push is made**.
- Your files must be committed to the **master branch**. Otherwise, it will not be scored.
- Your program should output correct results even for inputs other than those used in the example.
- Basically, assignments are scored based on the output results. If it is not possible to check whether a requirement is implemented because the output is not correct, no score is given for the requirement, even if it is implemented internally. However, even if the output result is correct, no score is given for a requirement if the internal implementation does not satisfy the requirement.

1. Write a program that works as follows:
  - A. Class Animal has two member variables, std::string name and int age, and has a constructor that initializes the values of name and age by taking a string and an integer as arguments.
  - B. Class Zebra that inherits Animal has a member variable, int numStripes, and has a constructor that initializes the values of name, age, and numStripes by taking a string, an integer, and an integer as arguments.
    - i. The member variable of the parent class is initialized through the constructor of the parent class.
  - C. Class Cat that inherits Animal has a member variable, std::string favoriteToy, and has a constructor that initializes the values of name, age, and favoriteToy by taking a string, an integer, and a string as arguments.
    - i. The member variable of the parent class is initialized through the constructor of the parent class.
  - D. Each class has a member function printInfo()
    - i. Animal::printInfo() does nothing.
    - ii. Zebra::printInfo() and Cat::printInfo() print out the type, name, age, and number of stripes (or favorite toy) as shown in the following example.
- E. Create Zebra or Cat objects according to user input, and put them into std::vector<Animal\*> animals.
- F. When the user enters 0, call the printInfo() function of each element of animals. Each element of animals must be deallocated after use.
- G. Do not use the type casting operator throughout the code.
- H. This program should take user input repeatedly
  - I. **Input:**
    - i. 'z' [name] [age] [numStripes] – Create a zebra
    - ii. 'c' [name] [age] [favoriteToy] – Create a cat
    - iii. 0 – Stop taking inputs and call printInfo() functions..
- J. **Output:** The result for calling printInfo() functions

K. Files to submit:

- i. main.cpp – main() must be in this file.
- ii. animal.h – Class definitions
- iii. animal.cpp – Class member function definitions (implementations)
- iv. A CMakeLists.txt to generate the executable

```
$ ./Animals
z Tom 2 21
z Amy 3 22
c Kitty 4 mouse
c King 3 tower
z Elen 5 30
0
Zebra, Name: Tom, Age: 2, Number of stripes: 21
Zebra, Name: Amy, Age: 3, Number of stripes: 22
Cat, Name: Kitty, Age: 4, Favorite toy: mouse
Cat, Name: King, Age: 3, Favorite toy: tower
Zebra, Name: Elen, Age: 5, Number of stripes: 30
$
```

2. Write a program for drawing multiple 2D shapes.

A. Complete the following classes using inheritance.

```

class Canvas {
public:
    Canvas(size_t row, size_t col);
    ~Canvas();

    // Change this canvas size to w x h. Preserve already existing shapes
    // on the canvas.
    void Resize(size_t w, size_t h);

    // Dot with the brush at (x,y). If (x,y) is outside the canvas, just
    // ignore it.
    bool DrawPixel(int x, int y, char brush);

    //Print out the canvas
    void Print();

    // Erase all (initialize with '.')
    void Clear();
private:
    // Define data member to save drawn shapes (note the resize function)
};

class Shape {
public:
    virtual ~Shape();
    virtual void Draw(Canvas* canvas) {};
protected:
    // Define common properties of shapes
};

class Rectangle : public Shape { /* Define required members */ }
class UpTriangle : public Shape { /* Define required members */ }
class DownTriangle : public Shape { /* Define required members */ }
class Diamond : public Shape { /* Define required members */ }

```

B.

### C. Note

- i. Take canvas size (width, height) from the user first.
- ii. Take user input repeatedly to create different objects of child classes of class Shape as shown in the following example.
- iii. Once an object is created, store it in a std::vector<Shape \*> object. All shape objects should be stored in the single vector object.
- iv. Redraw all shapes whenever displaying the canvas.
- v. All shapes in the canvas must be able to 'overlay'. The shape entered later overwrites the shape entered earlier.
- vi. Empty spaces are printed with '.' and spaces in the shape are printed with brush characters.

D. This program should take user input repeatedly

E. **Input:**

- i. 'add' [shape] [shape parameters]
  1. 'rect' [top-left x] [top-left y] [width] [height] [brush]
  2. 'diamond' [top-center x] [top-center y] [distance from center to each corner] [brush]
  3. 'tri\_up' [top-center x] [top-center y] [height]
  4. 'tri\_down' [bottom-center x] [bottom -center y] [height]
- ii. 'delete' [shape index] – Delete a shape
- iii. 'draw' – Draw the canvas
- iv. 'dump' – Print out information about all shapes in the vector.
- v. 'resize' [new width] [new height] – Resize the canvas
- vi. 'quit' – Quit the program.

F. **Output:** The result for each command.

G. Files to submit:

- i. main.cpp – main() must be in this file.
- ii. canvas.h – Class definitions
- iii. canvas.cpp – Class member function definitions (implementations)
- iv. A CMakeLists.txt to generate the executable

```
$ ./draw_shape
10 10
0123456789
0.....
1.....
2.....
3.....
4.....
5.....
6.....
7.....
8.....
9..... // Draw a empty canvas once when you first create the canvas
add rect 4 4 3 3 *
draw
0123456789
```

```

0.....
1.....
2.....
3.....
4....***...
5....***...
6....***...
7.....
8.....
9..... // Draw a rectangle of width 3 and height 3 with (5, 5) at top
left
add tri_down 3 3 3 @
draw
0123456789
0.....
1.@@@@@.....
2..@@@.....
3...@.....
4....***...
5....***...
6....***...
7.....
8..... // Draw an inverted triangle of height 3 with (3, 3) an the
bottom center
9.....
add tri_up 5 7 3 #
draw
0123456789
0.....
1.@@@@@.....
2..@@@.....
3...@.....
4....***...
5....***...
6....***...
7....#
8....##...
9...####.. // Draw a triangle of height 3 with (7, 7) an the top center
add diamond 2 5 2 ?
draw
0123456789
0.....
1.@@@@@.....
2..@@@.....
3...@.....
4....***...
5..?..***...
6.???***...
7?????#.....
8.???##...
9..?###.. // Draw a diamond with (2, 5) at top center, having distance 2
from center to each corner

add rect 5 5 8 4 +
// Shapes are only drawn inside the canvas
dump
0 rect 4 4 3 3 *
1 tri_down 3 3 3 @

```

```

2 tri_up 7 7 3 #
3 diamond 2 5 2 ?
4 rect 5 5 8 4 +
draw
0123456789
0.....
1.@@@@@.....
2..@@@.....
3...@.....
4....***....
5..?.*+++++
6.???*+++++
7?????+++++
8.???#+++++
9..?####..
delete 5 // Ignore deleting invalid index
delete 0
dump
0 tri_down 3 3 3 @
1 tri_up 7 7 3 #
2 diamond 2 5 2 ?
3 rect 5 5 8 4 +
draw
0123456789
0.....
1.@@@@@.....
2..@@@.....
3...@.....
4.....
5..?.+////
6.???.+////
7?????+////
8.???#+////
9..?####..
resize 15 10 // Resize the canvas
draw
012345678901234
0.....
1.@@@@@.....
2..@@@.....
3...@.....
4.....
5..?.+++++...
6.???.+++++...
7?????+++++...
8.???#+++++...
9..?####.....
// Now you can see the shape that couldn't come out because
of the canvas size
quit
$
```