SIT102 – Introduction to Programming

Answers for 1.2P Shape Drawing

Student Name: Yizheng HE
Student ID: 221411294

Question 1: Why do we create procedures? What are the advantages of doing this? How will this helps make it easier to build larger programs?

In a computer, we create procedures. A program is a separate code that performs a specific task. Programming, on the other hand, is instruction, which contains a set of instructions to be executed and can be understood as workers in a program that can be called to perform some task that can then be referenced in a larger procedure. The benefit of programming is that it can help build powerful database applications. Stored procedures have several advantages, including better performance, higher productivity, ease of use and increased scalability.
Procedures can be used throughout the program, making them easier and faster to code. There is an additional benefit to using stored procedures. If something needs to be changed in a procedure, it only needs to be changed once in the programming code. This change will appear anywhere in the program where the procedure is used.

Question 2: In what order will the computer execute (run) the instructions in your code? Where do these instructions start from?

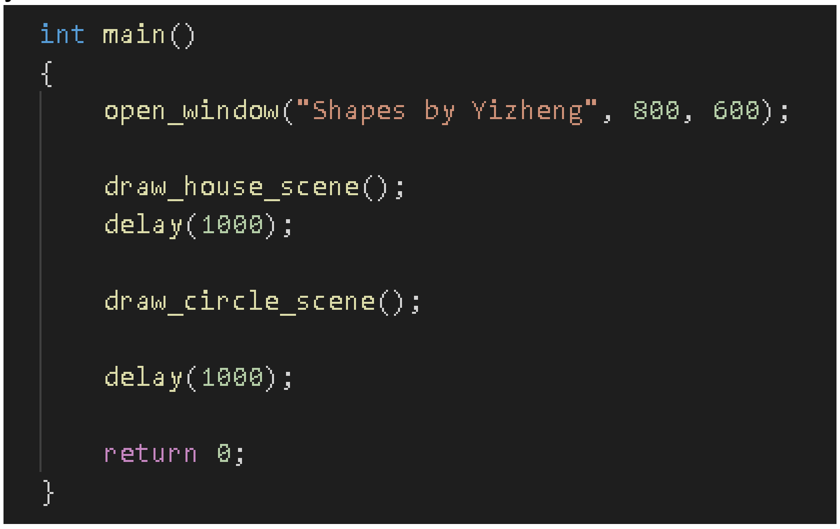
The computer will start with int main() {where the contents are executed in order}, for example, I wrote open_window("Shape by yizheng", 800,600) in the first sentence, that is,to open the window and name it Shape by Yizheng, the window size is 800×600.
Draw_house_scene, delay (1000); Draw_circle_scene, return 0; So the computer starts running instructions in priority order

Question 3: How do we "call a procedure" in a program? In addition, please give a code/syntax example of procedure call used in your 1.2P Shape Drawing program.

For example, THE draw_house_scene code did not work when I tried to draw a graphic. I learned from the video that this phrase needs to be defined before it can be used. Here it is defined using void draw_house_scene()

```
{
  clear_screen(COLOR_WHITE);
  fill_ellipse(COLOR_BRIGHT_GREEN, 0,400,800,400);
  fill_rectangle(COLOR_GRAY,300,300,200,200);
  fill_triangle(COLOR_RED, 250,300,400,150,550,300);
```

refresh_screen(60);
delay(5000);
}



```
int main()
{
    open_window("Shapes by Yizheng", 800, 600);

    draw_house_scene();
    delay(1000);

    draw_circle_scene();

    delay(1000);

    return 0;
}
```

when we
define draw_house_scene, then we can "call a procedure" in our program as if it were a pre-defined scene. void literally means empty type, void * means empty type pointer, void is not a real type, the real use of void is in the following two areas: compiler Qualifying the return value of a function for compilation
Qualification of function parameters class

Question 4: There are different (overloaded) version(s) defined for **fill_rectangle(…)** in SplashKit (https://www.splashkit.io/api/graphics/#group-fill-rectangle). With an aid of one of its available versions (please choose one), **elaborate** the definition of **fill_rectangle(…)** in regard to its input parameters. Give an example for **each input parameter** (what could be a **valid passed-in value**) respectively in a procedure call to it.

For example, when we use fill_rectangle(Color_Unknow, x,y,w,h); Assuming that we have set the length and width of the window to 800×600, I use fill_rectangle(COLOR_GRAY,300,300,200,200); What is the definition of input parameters for? First of all, the color of the graph is gray, the graph is 300 on the x-coordinate, 300 on the y-coordinate, 200 on the length and 200 on the width, which is similar to finding a point in a linear equation, but the y down here is not negative. X:the distance from the leftof the window/bitmap to the rectangle. Y:the distance from the top of the
window/bitmap to the rectangle.

-- End of questions (4) --