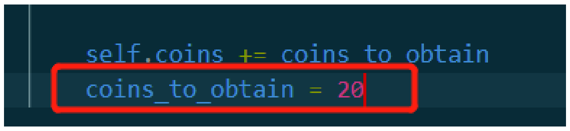
CSCI3180 HW3 Report

LI Jialu 1155107895

1. Provide example code and necessary elaborations for demonstrating the advantages of Dynamic Scoping in using Perl to implement the Advanced Tournament Dual game as compared to the corresponding codes in Python.

In Python, the I have to manually restore the value for those delta_attack attributes etc. It's not convenient and cause a lot of troubles of debugging.

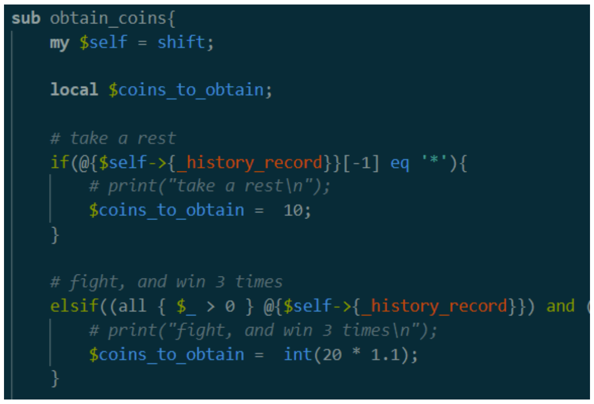


```
self.coins += coins_to_obtain
coins_to_obtain = 20
```

In Perl, these values will be automatically restored after leaving the scope use dynamic scoping.

I only have to declare a packaging variable with keyword "local", then perl automatically create a variable with a perfect scoping as I need.

So that I can pass their values to the function calls it and don't have to really touch the global variable's value.

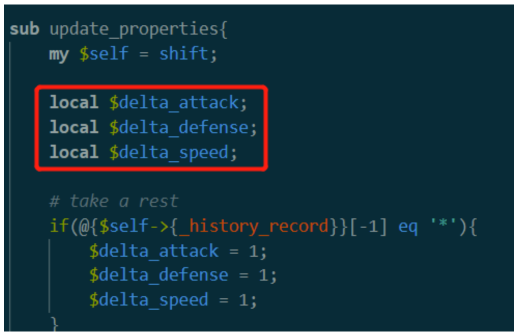


```
sub obtain_coins{
    my $self = shift;

    local $coins_to_obtain;

    # take a rest
    if(@{$self->{_history_record}}[-1] eq '*'){
        # print("take a rest\n");
        $coins_to_obtain =  10;
    }

    # fight, and win 3 times
    elsif((all { $_ > 0 } @{$self->{_history_record}}) and (
        # print("fight, and win 3 times\n");
        $coins_to_obtain =  int(20 * 1.1);
    }
```



```
sub update_properties{
    my $self = shift;

    local $delta_attack;
    local $delta_defense;
    local $delta_speed;

    # take a rest
    if(@{$self->{_history_record}}[-1] eq '*'){
        $delta_attack = 1;
        $delta_defense = 1;
        $delta_speed = 1;
    }
```

2. Discuss the keyword local in Perl (e.g. its origin, its role in Perl, and real practical applications of it) and give your own opinions.

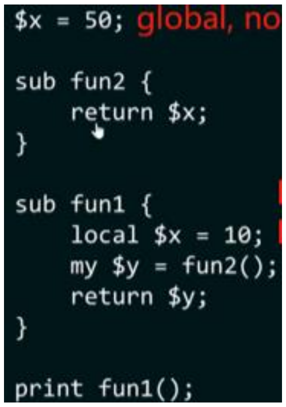
As for the origin of "local" keyword one possible explanation form Wikipedia (1) is below:

"A special type of local variable, called a static local, is available in many mainstream languages (including C/C++, Visual Basic, and VB.NET) which allows a value to be retained from one call of the function to another – it is a static variable with local scope. In this case, recursive calls to the function also have access to the (single, statically allocated) variable. In all of the above languages, static variables are declared as such with a special storage class keyword (e.g., static). "

Conclude the above, it says that "local" keyword's origin may be traced back to other programming languages e.g. C, and it's designed to differ from the "global" variable.

The role in Perl is mainly to give a temporary, dynamically-scoped value to a global variable. And it can just last in this certain enclosing block. However, the variable is visible to any function called from within the block.

Below is a real application of how "my" and "local" keyword" differ from each other.



```
$x = 50;

sub fun2 {
    return $x;
}

sub fun1 {
    local $x = 10;
    my $y = fun2();
    return $y;
}

print fun1();
```

In this code snippet above, we use "local $x", so that fun2 will trace back to this new value 10, instead of the old value 50 defined globally. The output will be 10. If we change the keyword to "my $x", then the output will be 50.

Reference

(1) https://en.wikipedia.org/wiki/Local_variable#:~:text=Local%20variables%20in%20Perl,-Perl%20supports%20both&text=%22.,called%20from%20within%20the%20block.