Utku Boran Torun / Section-01 / 21901898

Nested Subprogram Definitions

Since I have experience Dart before from previous homework, defining nested subprograms was easy for me. I wrote two subprograms each where one of them creates an integer variable and the other one takes a parameter and multiplies with itself.

Code Segment

```dart
// Nested subprogram in Dart
void foo()
{
  int x = 2;
  void bar()
  {
    int y = 5;
    print("y = $y, x = $x");
  }
  bar();
}
```

```dart
// call the nested subprogram
print("Call the nested subprogram");
foo();
```

Output

```
Call the nested subprogram
y = 5, x = 2
```

In here there is nested subprograms which are foo and bar. foo calls the bar function and bar function just prints the x and y variables. As we can see from the output bar function can print both x and y.

Scope of Local Variables

From my point of view, in terms of writing and design, Dart and Java are alike. For this section, I can say the same thing as well. While implementing this, it was so intuitive, and I thought the output, without running the code, and it was true.

Code Segment

```
// Scope of the local variables
void foofoo()
{
  String a = "this is a";
  void barbar()
  {
    String b = "this is b";
    print("$a and $b");
  }
  barbar();
  //print(b); // this is error because b is not in the scope of the foofoo
}
```

```
  int a = 4;
  {
    a = 2;
    {
      int b = 3;
    }
    //print(b); // error, b is not in the scope
  }
  print("a is $a");
  //print(b); // error, b is not in the scope
```

Output

```
Testing the scope of local variable
this is a and this is b
a is 2
```

In this part, I examined the scope of local variables in both functions and blocks. As we can see from the first code segment, there is string variables named as ‘a’ and ‘b’ in the functions. The variable a is declared in the function foofoo and b is declared in function barbar and when I try to print both a and b in barbar, it is successful. On the other hand, when I try to print b in the foofoo it is an error. The same thing applies in blocks as well the integer variable a is declared outside of the blocks and changed in the outer block, then in the inner block integer b is declared, and it is assigned to 3. When I tried to print a and b outside the inner block, it is an error.

Parameter Passing Methods

In Dart there is no pass-by-reference, there is just a pass-by-value. In this homework, I examined pass-by-value by normal parameters and functions.

Code Segment

```
// parameter passing methods

// pass by value
void addTen(int a)
{
  a = a + 10;
}

// passing functions
void boran(void Function(int) f, int a)
{
  f(a);
}
```

```
// function that is passed as parameter in the function boran
void multiplyBy2(int a)
{
  a = a * 2;
  print( "In multiply by 2, a is $a");
}
```

```
// testing parameter passing methods
print("Testing parameter passing methods");

int number = 4;
addTen(number);
print("Number is $number"); // we expect 14, but it prints 4 because it pass by value

boran(multiplyBy2, 34);
```

Output

```
Testing parameter passing methods
Number is 4
In multiply by 2, a is 68
```

This code creates a method named boran and takes its parameters as function and integer. This boran function does not do anything, but it calls the function which is passed as a parameter. Then it creates another method called multiplyBy2 and it takes one parameter which is an integer. This method multiplies the parameter with 2 and prints it. After that, this code creates addTen function which adds 10 to the passed argument, but actually, it does not add 10, because it is a pass-by-value. With doing this, pass-by-value is examined. After this main method calls these functions and prints their output.

Keyword and Default Parameters

In Dart, there are three types of parameters, these are default, optional, and named. In this homework, I examined all three of them. It was hard to understand what a keyword is, but then I understood.

Code Segment

```dart
// keyword and default parameter part
void testDefaultParameter([int a = 100])
{
  print("In default parameter a = $a");
}

// named parameter
void testNamedParameter({required String s, int a = 43 })
{
  print("$s and a is $a");
}
```

```dart
// keyword and default parameter
print("Testing default parameter");
testDefaultParameter();
testDefaultParameter(15);

print("");

print("Testing named parameter");
testNamedParameter(s : "This is s");
testNamedParameter(s : "This is s", a : 5);
```

Output

```
Testing default parameter
In default parameter a = 100
In default parameter a = 15

Testing named parameter
This is s and a is 43
This is s and a is 5
```

This code creates two functions that test the default parameter and the named parameter. In the first one optional parameter is used with default values, so if the parameter is not passed in it, integer a will be equal to 100 always. On the other hand, if a parameter is passed, a will take that value. In the second method, the keyword ‘required’ is used. This keyword specifies that you cannot call the function without passing a parameter in it. In the main function, this function is called with both default parameters and without default parameters. As in the first function if integer a is specified in passing parameter, a takes that value, but if it is not specified a is always 43.

Closures

Before this homework, I did not have any experience with closures, but I knew what it was. I did little research on the official document of Dart and find out how closures are implemented in Dart.

Code Segment

```
Function multiply(int multiplyBy)
{
  return (int i) => multiplyBy * i;
}
```

```
// closures
var x = multiply(5);
print("Testing closures in Dart, by multiplying x by 4, x = " + x(4).toString());
```

Output

```
Testing closures in Dart, by multiplying x by 4, x = 20
```

This code creates a closure in Dart and tests it. The Function multiply takes a parameter as an integer, and it multiplies with the given parameter. Then it creates a var in the main method by using closure and putting 5 as a parameter, and now x is equal to the multiply closure with 5. When it is called in the print function, 4 is passed as a parameter, and it is equal to 5 * 4 which is 20.

Evaluation of The Language

As I have experience with Dart in previous homework, I knew Dart in terms of writability and readability. I will evaluate all design issues one by one

- Nested Subprogram Definitions

This section is one of the easiest parts of this homework, because Dart is good at language to code nested subprogram definitions in terms of writability and readability. In addition, this part was so intuitive to implement, because it is similar to Java's nested subprogram definitions.

- Scope of Local Variables

As I said in the previous section, again, Dart is good for the testing scope of local variables in terms of writability and readability, because this part is similar to Java as well. When I was writing this part, I didn't need to look document of the Dart, and it was intuitive as well.

- Parameter Passing Methods

When I searched for parameter passing methods in Dart, I learned that there is no pass-by-reference, but there is pass-by-value in Dart. First I tried myself and it was, so intuitive to do it. In terms of writability and readability, Dart is good at parameter passing methods.

- Keywords and Default Parameters

From my point of view, this was the hardest part of the homework, because first I did not understand what 'keywords' mean for parameters. After this, when I did little research, I understood and coded after that. In terms of readability, Dart is good at keywords and default parameters, on the contrary, it is not good in terms of writability, because when I tried to implement optional parameter, I needed to use '[]' (square brackets], but when I tried to use named parameters, I needed to use '{}' (curly brackets). I think this decreases the writability of the language.

- Closures

I think closures were one of the easiest parts of the homework after nested subprograms and the scope of local variables. I fastly found out how to implement closures from the documents and then tested it. In terms of readability and writability Dart is good at closures.

Learning Strategy

My main learning strategy was mainly doing research on the Internet. I used two sources which are StackOverFlow and the official website of Dart. First, I tried to understand what the assignment was and what the assignment required to finish it. After this, I understood every section in terms of what they are, and after understanding what they are, I searched for the implementation of every part. At first, I looked at the official documentation of Dart, and if I couldn't find it there, I searched on the internet, and StackOverFlow was sufficient to finish the assignment. After doing all these, homework was finished. The online compiler that I used,

Dart: https://dartpad.dev/

References

1-https://dart.dev/guides/language/language-tour

2-https://stackoverflow.com/