CS315 HOMEWORK 3 REPORT

Onurcan Ataç
22002194
Section 2

Implementation

Explanations of the example programs are made by detailed comments on code part. Different mini programs in codes are separated with header with respect to homework parts given. Online compiler used is https://dartpad.dev/.

Code

```
void main() {
  //Nested SubProgram Definitions
  print("");
  print("Nested SubProgram Definitions");
  print("");

  //Inner functions (function definitions inside function
definitions) are supported in Dart, and they can be called by using
the outer function.

  void function1(int number, String word, double float) {

    void insideFunction1(int number) {
      int innerNumber = number + 15;
      print("Sum is $innerNumber \n");
    }

    void insideFunction2(String word) {
      String innerString = "$word is a word \n";
      print(innerString);
    }

    void insideFunction3(double float) {
      double innerFloat = float + 3.1;
      print("Float is $innerFloat\n");
    }

    insideFunction1(number);
    insideFunction2(word);
    insideFunction3(float);
  }

  function1(7, "Two ", 5.2);

  //Scope of Local Variables
  print("");
  print("Scope of Local Variables");
  print("");
  //Dart is a lexically scoped language.
  //Variables in the functionOut can only be accessed from
functionOut, while variables in functionIn can be
  //accessed from both functions.

  //functionOut can't reach string3

  void functionOut(String s1, s2) {
    String string1 = s1;
    String string2 = s2;

    print("string1 = $string1 \n");
    print("string2 = $string2 \n");


   //functionOut cannot react string3, so if string3 would be
printed here, it would result with an error.

    void functionIn(String s3) {

      String string3 = s3;

      print("string1 = $string1 \n");
      print("string2 = $string2 \n");
      print("string3 = $string3 \n");
    }

    functionIn("Bilkent");

  }

  functionOut("Onurcan", "Ataç");

  //Parameter Passing Methods
  print("");
  print("Parameter Passing Methods");
  print("");

  //Dart uses pass by value. Dart does not use pass by reference for
primitive types.

  String aString = "What is my name?";

  void answer(String question) {
    question = "$question My name is Onur.";
    print(question);
  }
  answer(aString);
  //Since Dart uses pass by value, if aString is displayed, it
wouldn't show the changes which are done in the function.
  print(aString);


  //Arrays are passed with pass by value as well, as can be seen
below.
  var names = ["Tate", "Andrew", "Messi"];

  void addNames(var array)
  {
    array = "Ronaldo";
    print(names);
  }

  addNames(names);

  //array will not be changed to Ronaldo since Dart uses pass by
value.
  print(names);


  //Keyword and Default Parameters.
  print("");
  print("Keyword and Default Parameters");
  print("");

  //Dart supports required and optional parameters. Optional
parameters should have default values since the
  //user does not have to provide values for optionals while calling
the function.

  //firstFunc has required parameters and they need to be provided
in order to work properly. If the
  //method is called without providing any of the values program
will result in a compile error.
  int firstFunc(int var1, int var2, int var3) {
    return var1+var2+var3;
  }

  //There are two types of optional parameters in Dart. Positional
and named optional parameters.
  //All optional parameters have to have a default value since a
value might not be given to them while
  //calling the function

  //no2 is a positional optional parameter, it has default value 7.
  int secondFunc(int no1, [int no2 = 7]) {
    return no1 + no2;
  }

  //Providing both parameters is the first option, default value of
no2 will be avoided and no2 will
  //be taken as 4. no1 is 2.
  print(secondFunc(2, 4));
  //Not providing both parameters is the second option, default
value of no2 will be taken as and no2 will
  //be taken as 4. no1 is 2.
  print(secondFunc(14));

  //in positional parameters, the order is important. The function
below has 2 positional parameters, so we have to be careful with the
passed argument order.
  int thirdFunc(int var1, [int var2 = 7, int var3 = 13, int var4 =
43]) {
    return var1+var2+var3+var4;
  }

  print(thirdFunc(2));
  print(thirdFunc(2,4,8));


  //Order of the named optional parameters does not matter since
they are given with a label while
  //the function is being called. They are still optional and has
default values.

  //no2, no3 and no4 are named optional parameters.
  int fourthFunc(int var1, {int var2 = 5, int var3 = 7, int var4 =
11}) {
    return var1+var2+var3+var4;
  }

  print(fourthFunc(1));
  print(fourthFunc(1, var3: 16));
  print(fourthFunc(1, var3: 17, var2: 31, var4: 54));

  //Closures.
  print("");
  print("Closures");
  print("");

  //Closures are structures similar to functions, but they don't
have names, they can return values but they don't always do.
  //There might be statements in the scope of closures.
  //They can be called directly after definition as in the
following.
  (String name)
  {
    print("$name is a CS student.");
  }("Onurcan");

  //Variables can be declared as aliases for closures. By this way,
they can
  //be called more conveniently.

  var printString = (String name, String department)
  {print("$name is a $department student.");};

  printString("Onurcan", "CS");

  //Closures can also be written in arrow syntax.

  var printStringOther = (String department) => print("Onurcan is a
$department student.");
  printStringOther("CS");
}
```

Output:

```
Nested SubProgram Definitions

Sum is 22

Two  is a word

Float is 8.3


Scope of Local Variables

string1 = Onurcan

string2 = Ataç

string1 = Onurcan

string2 = Ataç

string3 = Bilkent
```

Parameter Passing Methods

```
What is my name? My name is Onur.
What is my name?
[Tate, Andrew, Messi]
[Tate, Andrew, Messi]
```

Keyword and Default Parameters

```
6
21
65
57
24
33
103
```

Closures

```
Onurcan is a CS student.
Onurcan is a CS student.
Onurcan is a CS student.
```

Evaluation of Language

Dart was already a language I liked coding in homeworks, it is fairly readable and writable, while also presenting a variety of options for programmers to use. For an example, required and optional parameters are great features in my opinion. Even though they are not a must in order to write methods and by including them a sacrifice from readability is made, I think those functionalities add more to the language. I didn't see anything taking away from readability and writability without giving more in functionality. Hence, in my opinion, there is no wasted readability or writability in Dart.

Learning Strategy

For this task, I thought that there might be enough explanation on the internet about subjects covered in the homework. I used internet sites as sources to try things out in online compiler, and I wrote the code in the file in result. I completed the homework repeating this routine for each language. I experimented on the online compilers to check if some syntax that might work, using my experience from other programming languages. I also checked documentation of the language where I got stuck.

References

"Dart programming language," *Dart*. [Online]. Available: https://dart.dev/. [Accessed: 17-Dec-2022].

"Dart closures tutorial with examples," *o7planning*. [Online]. Available: https://o7planning.org/14061/dart-closures. [Accessed: 17-Dec-2022].

"How to create an array in Dart," *Educative*. [Online]. Available: https://www.educative.io/answers/how-to-create-an-array-in-dart. [Accessed: 17-Dec-2022].

"Dartpad," *DartPad*. [Online]. Available: https://dartpad.dev/. [Accessed: 17-Dec-2022].