B2.3

# Programming constructs

CHAPTER ASSIGNMENT

Name _____ Class _____ No. _____ Date _____

Answer every question in the space given. Show your working. You may use the chapter page to help you.

Questions from *Computer Science for the IB Diploma*, 1st edition (2025), Paul Baumgarten, Ioana Ganea, Carl Turland (Hachette Learning), for classroom use.

## Programming Exercises — Selection Structures (p. 318)

1. Construct code to let the user input their favourite food. If the user enters `pasta`, `PASTA` or `Pasta`, output the message `"Go to Italy"`. Otherwise, output the message `"[favouriteFood] will never replace pasta!"`

2. Construct code that requires the user to input a number and a Boolean variable `flag`. If the `flag` is true, output double the number. Otherwise, output the squared number.

3. Retirement ages differ for men and women. In some countries, women can retire at the age of 62 and men at 65. Construct a program that asks the user to enter their name, gender and age. Calculate and output whether the user is eligible for retirement, or how many years they need to wait to get their pension.

4. Construct a program that requires the user to enter the measurements of the sides of a right-angled triangle. Calculate and display its area.

5. Construct code to create a calculator that converts temperatures between Celsius and Fahrenheit.

6. Construct code to create a quiz that asks the user to guess the capital city of five different countries of your choice. Add a score to your quiz. At the end, display the right answers for the missed questions and the score.

7. Construct a program that asks the user to enter their favourite colour and their lucky number. If their favourite colour is green and their lucky number is between 4 and 7, output the message `"In [LuckyNumber] years, you'll buy a [colour] bicycle"`. Otherwise, if the lucky number is below 4 and above 1, output the message `"I suppose you wanted a [colour] ball [LuckyNumber] years ago"`. Otherwise, display the message `"[LuckyNumber] is not so lucky!"`

8. Construct code that requires the user to input two numbers. Compare the two numbers and output the larger one. If the two numbers are the same, output an appropriate message instead.

## Review Questions — Loops and Iteration (p. 325)

1. Define the term "loop".

2. Explain why it is useful to use loops in coding.

3. Explain the difference between a pre-condition and a post-condition loop.

4. Identify a situation when you would use a count-controlled loop.

5. Define the term "termination condition".

6. Explain why it is important to avoid infinite loops.

7. Identify a situation where an infinite loop could occur.

## Programming Exercises — Loops and Iteration (p. 325)

1. Construct a program that allows the user to enter a number. Output all numbers between 1 and that number.

2. Construct code that requires the user to enter a number and a word. Display each letter in the word on a separate line. Repeat the process the user's number of times.

3. Construct code that asks the user to input ten numbers. Calculate and display the sum of all even numbers entered.

4. Construct a program that creates a variable and initializes it to the value 7. Ask the user to guess a number between 1 and 20. If the user guesses the value 7, output the number of attempts. Do not allow the user to try to guess the number more than three times. If the user doesn't guess the number after three attempts, output the message: `"The game is locked. Try again later!"`

5. Construct a program that asks the user whether they want an apple. Repeat the question until their answer is yes. At the end, output the message: `"Apples are delicious, aren't they? Let's have one!"`

## Review Questions — Functions and Modularization (p. 329)

1. Compare functions and procedures.

2. Identify three things you can do to keep a maintainable code.

3. Outline the difference between validation and verification. Provide an example where a selection can be used for validation checks.

4. Differentiate between pre- and post-condition loops.

### Programming Exercises — Functions and Procedures (p. 329)

1. Construct code to let the user create a function called `factorial` that takes a natural number (`n`) as a parameter. The function should return the factorial value of that number ($n!$).

2. Construct a program that consists of a procedure called `odds` that takes a natural number (`n`) as a parameter. The procedure should display all the odd numbers from 1 to that number (`n`).

3. Construct code that creates a function called `perfect` that takes `n` (a natural number) as a parameter. The function should return true/false if `n` is a perfect number. A perfect number is a positive integer that is equal to the sum of its positive proper divisors, excluding the number itself. For example, 28 is a perfect number as its proper divisors are: 1, 2, 4, 7, 14. Adding those values together we get 28, and this result is equal to the initial number.

4. Construct a function called `palindrome` that takes a word as a parameter. The function should return true/false if the given word is a palindrome. A palindrome is a word that reads the same forwards or backwards. For example, `kayak`, `racecar`, `level` and `civic` are all palindromes.

## Review Questions — Scope of Variables (pp. 331–332)

1. Trace the following program to determine its output.

```
static String student = "Bob";
public static void changeOfName(String st) {
    st = "Bobby";
    System.out.println("Student inside the method: " + st);
}
public static void main(String[] args) {
    changeOfName(student);
    System.out.println("Student outside the method: " + student);
}
```

```
student = "Bob"
def changeOfName(st):
    st = "Bobby"
    print("Student inside the method: ", st)
changeOfName(student)
print("Student outside the method: ", student)
```

2. Identify a local variable in the code above. Explain why the chosen variable is not a global variable.

3. Construct a program that changes the code above so that, after the method is executed, both outputs display the name `Jim`, without changing the first line of code.

4. Construct code to update the method above to include a validation check: the new name should be input from the keyboard and the change should occur only if the new name is different from the previous name. Display a message if the new name is not different and display the current name.

5 Identify the scope and data type of the variable you have created to validate the name change.

End of assignment · Check your answers against the chapter at gurujazzy.com/subjects/ib-computer-science/b2-3-constructs/