📘 C++ for Clever Kids – Exercise Book

✨ *Chapter 1: Hello, Computer!*

Exercise 1: Say Hello! (Single & Multiple Choice)

1. What does `std::cout` do?
   a) Add numbers
   b) Show something on screen
   c) Wait for typing
   d) Close the program

2. What goes at the end of a C++ statement?
   a) `:`
   b) `.`
   c) `;`
   d) `!`

3. Which one is the correct main function?
   a) `main() int {`
   b) `int main {`
   c) `int main() {`
   d) `start main()`

What will this print?

```
std::cout << "Hello";
```

4. a) Hello
   b) hello
   c) "Hello"
   d) Nothing

5. Which keyword ends the program?
   a) `stop`
   b) `end`
   c) `return`

d) `finish`

Exercise 2: Fix My Code (Fill-in-the-Blank)

Fill in the missing pieces in this C++ program:

```
#include <________>          // [1]
________ main() {           // [2]
    std::cout << "Hi!";     // [3]
    ________ 0;             // [4]
}                           // [5]
```

1. _____
2. _____
3. (No change)

4. _____
5. _____

Exercise 3: What Will It Print? (Single Choice)

1. `std::cout << "Hello\nWorld!";`
   a) HelloWorld!
   b) Hello World!
   c) *(Hello on one line, World! on the next)*
   d) Error

2. `std::cout << "I love C++!";`
   a) I love C++!
   b) I love C + +
   c) Error
   d) I love C

3. `std::cout << "123" << "456";`
   a) 123456
   b) 579

c) 123 456
d) "123""456"

4. `std::cout << "Coding is fun!\n";`
   What does `\n` do?
   a) Makes a beep
   b) Adds a space
   c) Ends the program
   d) Goes to new line

5. Which one prints *nothing*?
   a) `std::cout << "";`
   b) `std::cout << " ";`
   c) `std::cout << "Hi";`
   d) `std::cout << "\n";`

Exercise 4: Let's Code a Bit (Coding Choice)

What should go in the blanks?

```
#include <iostream>

int main() {
    std::cout << "________, world!"; // [1]
    std::cout << "\n________!";      // [2]
    return 0;
}
```

1. Fill the blank to greet the world (e.g., "Hello")

2. Add your name in the second blank

✅ Answers: Chapter 1

Exercise 1

1. b
2. c
3. c
4. a
5. c

Exercise 2

1. `iostream`
2. `int`
3. —
4. `return`
5. `}`

Exercise 3

1. c
2. a
3. a
4. d
5. a

Exercise 4

1. `Hello`

2. Your name (e.g., `Leah`)

📘 C++ for Clever Kids – Exercise Book

✨ *Chapter 2: What Are Numbers (and Friends?)*

Exercise 1: Type It Right (Multiple Choice)

1. What type holds whole numbers like 1, 2, 3?
   a) `float`
   b) `std::string`
   c) `int`
   d) `char`

2. What type is used for letters like `'A'`?
   a) `char`
   b) `word`
   c) `int`
   d) `letter`

3. What type holds `true` or `false`?
   a) `yesno`
   b) `char`
   c) `bool`
   d) `flag`

4. Which type gives the **most** decimal places?
   a) `int`
   b) `float`
   c) `double`
   d) `string`

5. What does `const` mean?
   a) Big number
   b) Can change
   c) Always true

d) Cannot change

Exercise 2: Fill in the Type (Coding Fill-in-the-Blank)

Fill in the blanks with the correct C++ types:

```
________ age = 8;              // [1]
________ grade = 99.5f;        // [2]
________ pi = 3.14159;         // [3]
________ letter = 'A';         // [4]
const ________ magic = 42;     // [5]
```

Exercise 3: What's the Value? (Single Choice)

1. `int x = 5 + 2;`
   What is x?
   a) 6
   b) 7
   c) 8
   d) Error

2. `float f = 3.5 + 0.5;`
   a) 3
   b) 4
   c) 3.10
   d) 4.0

3. `char c = 'B'; std::cout << c;`
   a) 66
   b) B
   c) b
   d) Error

4. `bool happy = true;`
   What does `std::cout << happy;` print?
   a) true
   b) 1

c) yes
d) "true"

5. Which one is a **string**?
   a) `"Hello"`
   b) `'H'`
   c) `Hello`
   d) `H`

Exercise 4: Fix the Program (Coding Choice)

Fix the mistakes in this program:

```
int main() {
    std::string name = Hello;          // [1]
    float number = "9.8";              // [2]
    const int birth = "2015";          // [3]
    bool isHappy = yes;                // [4]

    std::cout << name << number;       // [5]
}
```

✅ Answers: Chapter 2

Exercise 1

1. c
2. a
3. c
4. c
5. d

Exercise 2

1. `int`
2. `float`
3. `double`
4. `char`
5. `int`

Exercise 3

1. b
2. d
3. b
4. b
5. a

Exercise 4

1. `"Hello"` (missing quotes)
2. `9.8f` (remove quotes)
3. `2015` (remove quotes)
4. `true` or `false` (not `yes`)
5. All good

📘 C++ for Clever Kids – Exercise Book

✨ *Chapter 3: Talking with the Computer* 👂💬

Exercise 1: Input or Output? (Single Choice)

1. What does `std::cin` do?
   a) Print text
   b) Get input from user
   c) End the program
   d) Make a list

2. What symbol is used with `cin`?
   a) `<<`
   b) `>>`
   c) `==`
   d) `//`

Which one asks for your name?

```
std::cout << "What's your name?";
std::cin >> name;
```

3. a) `std::cout`
   b) `std::cin`
   c) Both
   d) None

4. What happens after `std::cin >> age;`?
   a) Nothing
   b) Program ends
   c) You type something
   d) It prints "age"

5. Which is correct syntax?
   a) `cin << name;`
   b) `std::cin >> name;`
   c) `input >> name;`
   d) `cin input name;`

Exercise 2: Fill in the Gaps (Coding Fill-in-the-Blank)

```
#include <iostream>

int main() {
    std::string name;
    std::cout << "Enter your _______: ";   // [1]
    std::cin >> _______;                   // [2]
    std::cout << "Hi, " << name << "!";    // [3]
}
```

Exercise 3: What Will It Do? (Single Choice)

1. `std::cin >> num;`
   a) Prints a number
   b) Adds numbers
   c) Waits for you to type
   d) Shows nothing

If you type `"Sam"` into this program:

```
std::string name;
std::cin >> name;
std::cout << name;
```

2. What will it print?
   a) Sam
   b) name
   c) "Sam"
   d) Error

What is the result of this?

```
int a;
```

```
std::cin >> a;
```

3. a) Program crashes
   b) Waits for a number
   c) Skips to end
   d) Prints “a”

4. What is `std::endl` used for?
   a) End the program
   b) Make new line
   c) Clear input
   d) Pause

5. What happens if you enter a space in `std::cin >> name;`?
   a) It reads full name
   b) It stops at the space
   c) Error
   d) Prints the whole thing anyway

Exercise 4: Complete the Code (Coding Choice)

Fill in what’s missing in this mini conversation:

```
#include <iostream>

int main() {
    int age;
    std::cout << "How old are you? ";  // [1]
    std::________ >> age;             // [2]
    std::cout << "You are " << age << " years old.";  // [3]
    return 0;
}
```

✅ Answers: Chapter 3

Exercise 1

1. b
2. b
3. c
4. c
5. b

Exercise 2

1. `name`
2. `name`
3. already correct

Exercise 3

1. c
2. a
3. b
4. b
5. b

Exercise 4

1. already correct
2. `cin`
3. already correct

📘 C++ for Clever Kids – Exercise Book

✨ *Chapter 4: Lists of Things (Arrays!)*

Exercise 1: Array Basics (Multiple Choice)

1. What is an array?
   a) A list of numbers or items
   b) A number
   c) A loop
   d) A function

2. What does `marks[0]` refer to?
   a) The last number
   b) The second number
   c) The first number
   d) An error

3. What happens if you don't fill all the array values?
   a) Compiler error
   b) It crashes
   c) Empty ones become 0
   d) All values repeat

4. Which type is used to store an array of 5 whole numbers?
   a) `int arr(5);`
   b) `std::array<int, 5>`
   c) `int[] arr;`
   d) `array<int> arr;`

5. What goes inside the brackets `[ ]`?
   a) A letter
   b) A number index
   c) A function
   d) A class

Exercise 2: Fill in the Array (Coding Fill-in-the-Blank)

```cpp
#include <iostream>
#include <array>

int main() {
    std::array<int, 3> scores = {90, ____, 100};  // [1]
    std::cout << scores[0];  // [2]
    scores[1] = ____;        // [3]
    std::cin >> scores[2];   // [4]
    std::cout << scores[2];  // [5]
}
```

Exercise 3: What Will It Print? (Single Choice)

1. `std::array<int, 3> a = {1, 2, 3}; std::cout << a[1];`
   a) 1
   b) 2
   c) 3
   d) a[1]

2. `char letters[3] = {'C', 'P', 'P'}; std::cout << letters[0];`
   a) CPP
   b) 0
   c) C
   d) Error

3. `std::array<int, 3> arr = {7, 8}; std::cout << arr[2];`
   a) 8
   b) 0
   c) Error
   d) 7

4. Which of these prints every item in an array?
   a) `for (int i = 0; i < 3; ++i)`
   b) `while (i < arr)`
   c) `repeat 3`
   d) `int arr[3]`

5. What is the last valid index of a 5-element array?
   a) 5
   b) 6

c) 4
d) 0

Exercise 4: 2D Arrays (Coding Choice)

```
int table[2][3] = {
    {1, 2, 3},
    {4, 5, 6}
};

// What will this print?
std::cout << table[1][2];  // [1]

for (int i = 0; i < 2; i++) {
    for (int j = 0; j < 3; j++) {
        std::cout << table[i][j] << " ";  // [2]
    }
}
```

1. What is the value of `table[1][2]`?
2. What does the full loop print?
3. Which row has the value 5?
4. Which column is `table[0][2]` in?
5. How many total values are in this array?

✅ Answers: Chapter 4

Exercise 1

1. a

2. c
3. c
4. b
5. b

Exercise 2

1. 95 (or any int)
2. 90
3. 85 (or any int)
4. Input value
5. Input value printed

Exercise 3

1. b
2. c
3. b
4. a
5. c

Exercise 4

1. 6
2. 1 2 3 4 5 6
3. Row 1
4. Column 2

5. 6 total values (2 rows × 3 columns)

📘 C++ for Clever Kids – Exercise Book

✨ *Chapter 5: If This, Then That (Decisions!)*

Exercise 1: True or False? (Single Choice)

1. What does `if (a == b)` mean?
   a) a is bigger than b
   b) a is smaller than b
   c) a equals b
   d) a becomes b

2. What does `else` do?
   a) Ends the program
   b) Runs if `if` is false
   c) Repeats code
   d) Makes a new variable

3. What is `==` used for?
   a) Compare values
   b) Assign values
   c) Add numbers
   d) Show errors

4. Which symbol means “not equal”?
   a) `!=`
   b) `<>`
   c) `==!`
   d) `//`

What will this print?

```
int n = 4;
```

```
if (n % 2 == 0)
    std::cout << "Even";
else
    std::cout << "Odd";
```

5. a) Even
   b) Odd
   c) 2
   d) 4

Exercise 2: Fix the Code (Fill-in-the-Blank)

Fill in the blanks:

```
int age = 8;

if (age >= 10) {
    std::cout << "You're 10 or older!";
} ____ {
    std::cout << "You're younger than 10!";
}
```

1. What keyword goes in the blank? _____
2. Replace `10` with a number to make it always false: _____
3. Add another check with `else if (age == 9)` — where does it go? _____
4. Change the message to say "Almost there!" for age 9
5. Add a `return 0;` at the end — why?

Exercise 3: What Will Happen? (Multiple Choice)

1. `if (false) { std::cout << "Hi"; } else { std::cout << "Bye"; }`
   a) Hi
   b) Bye
   c) false
   d) Error

2. `if (score >= 90)` is true when score is:
   a) 80
   b) 70
   c) 90
   d) 60

Ternary operator:

```
int a = 3, b = 5;
std::cout << (a > b ? a : b);
```

3. What prints?
   a) 3
   b) 5
   c) a
   d) Error

4. What does `switch` help with?
   a) Repeating
   b) Input
   c) Picking one of many
   d) Loops

5. In a switch, `case 2:` means:
   a) If 2 is true
   b) If something equals 2
   c) Loop 2 times
   d) Print 2

Exercise 4: Fill the Switch (Coding Choice)

```
int num = 3;
```

```
switch (num) {
    case 1: std::cout << "One"; break;
    case 2: std::cout << "Two"; break;
    case ___: std::cout << "Three"; break;     // [1]
    default: std::cout << "Other";             // [2]
}
```

1. Fill in the missing case number
2. What will it print?
3. What happens if `break;` is missing?
4. Add a `case 4:` for “Four”
5. Can you use `string` in switch?

✅ Answers: Chapter 5

Exercise 1

1. c
2. b
3. a
4. a
5. a

Exercise 2

1. `else`
2. Any number below 10 (e.g., 5)

3. Between `if` and `else`
4. `else if (age == 9) std::cout << "Almost there!";`
5. It ends the `main()` function properly

Exercise 3

1. b
2. c
3. b
4. c
5. b

Exercise 4

1. `3`
2. `Three`
3. It prints the next case(s) too (fall-through)
4. `case 4: std::cout << "Four"; break;`
5. No — switch only works with `int`, `char`, or `enum` types

📘 C++ for Clever Kids – Exercise Book

✨ *Chapter 6: Let's Repeat (Loops!)*

Exercise 1: Loop Logic (Multiple Choice)

1. Which loop checks the condition *before* running?
   a) `do-while`
   b) `if`
   c) `while`
   d) `repeat`

2. Which loop runs *at least once*, no matter what?
   a) `for`
   b) `while`
   c) `do-while`
   d) `if`

What will this print?

```
for (int i = 0; i < 3; i++) {
    std::cout << i;
}
```

3. a) 012
   b) 123
   c) 345
   d) Error

4. What does `continue` do inside a loop?
   a) Stops everything
   b) Jumps to next round
   c) Ends loop
   d) Repeats the same value

5. Which loop is infinite?
   a) `while (false)`
   b) `for (int i = 0; i > 10; i++)`
   c) `while (true)`
   d) `do { break; } while (false);`

Exercise 2: Fix the Loops (Coding Fill-in-the-Blank)

```
// Print numbers from 1 to 5
int i = 1;
______ (i <= 5) {      // [1]
    std::cout << i;    // [2]
    i++;               // [3]
}


// Use a for loop
___ (int j = 0; j < 3; j++) {   // [4]
    std::cout << "Hi!";         // [5]
}
```

Exercise 3: What Will It Print? (Single Choice)

1. `while (i < 3) { i++; std::cout << i; }`, starting with `i = 0`
   a) 012
   b) 123
   c) 234
   d) 345

What does this print?

```
for (int i = 0, j = 2; i < 3; i++, j--) {
    std::cout << "i=" << i << ",j=" << j << "; ";
}
```

2. a) i=0,j=2; i=1,j=1; i=2,j=0;
   b) i=2,j=0; i=1,j=1; i=0,j=2;
   c) i=0,j=0; i=1,j=1; i=2,j=2;
   d) Error

What prints odd numbers only?

```
for (int i = 0; i < 10; i++) {
    if (i % 2 == 0) continue;
```

```
    std::cout << i;
}
```

3. a) 02468
   b) 13579
   c) 123456789
   d) 11111

```
std::string word = "Hi";
```
What does this print?

```
for (char c : word) {
    std::cout << c << " ";
}
```

4. a) H i
   b) Hi
   c) H-i
   d) Error

5. What does `break;` do in a loop?
   a) Makes a new line
   b) Jumps to the start
   c) Ends the loop
   d) Ends the program

Exercise 4: Loop Challenge (Coding Choice)

```
// Complete the code to count down from 3 to 1

for (int i = 3; i >= 1; ______) {     // [1]
    std::cout << i << " ";           // [2]
}

// Make a do-while loop that prints "Go!" once

int x = 0;
```

```
do {
    std::cout << "Go!";              // [3]
} ______ (x < 0);                   // [4]
```

1. Fill in missing update in for loop: _____
2. How many times does it print?
3. What will the do-while always do, even if x < 0 is false?
4. Can a loop run 0 times?
5. When should you use `do-while` instead of `while`?

✅ Answers: Chapter 6

Exercise 1

1. c
2. c
3. a
4. b
5. c

Exercise 2

1. `while`
2. already correct
3. already correct
4. `for`

5. already correct

Exercise 3

1. b
2. a
3. b
4. a
5. c

Exercise 4

1. `i--`
2. 3 times
3. Prints "Go!" once
4. Yes (e.g., `while (false)`)
5. When the loop must run at least once

📘 C++ for Clever Kids – Exercise Book

✨ *Chapter 7: Make Your Own Magic (Functions!)*

Exercise 1: Function Basics (Single Choice)

1. What is a function?
   a) A number
   b) A loop
   c) A mini-program

d) A string

2. What keyword defines a function?
   a) `define`
   b) `do`
   c) `int`
   d) `return`

What does this return?

```
int add(int a, int b) {
    return a + b;
}
```

3. a) a
   b) b
   c) a + b
   d) a - b

4. Can a function return nothing?
   a) Yes, with `void`
   b) Only if it's a loop
   c) No
   d) Only with `int`

5. What's a good reason to use a function?
   a) To make a number
   b) To stop code
   c) To reuse code
   d) To create arrays

Exercise 2: Fill in the Function (Coding Fill-in-the-Blank)

```
// A function that multiplies two numbers
int multiply(int a, int b) {
    return a * ____;      // [1]
}
```

```
// Call it inside main
int main() {
    int result = ________(4, 5);  // [2]
    std::cout << result;          // [3]
}
```

Exercise 3: What Will It Do? (Single Choice)

1. `void greet() { std::cout << "Hi!"; }`
   What type of function is this?
   a) Returns an int
   b) Returns a string
   c) Returns nothing
   d) Returns void

What happens here?

```
int square(int x) {
    return x * x;
}
std::cout << square(3);
```

2. a) 3
   b) 9
   c) 6
   d) Error

3. What happens if a function doesn't return anything?
   a) Nothing prints
   b) It breaks
   c) It must be `void`
   d) You get a warning

4. Which one is correct?
   a) `fun[int x]`
   b) `int fun(x)`
   c) `int fun(int x)`

d) `int = fun(x)`

5. Overloaded functions can:
   a) Only work once
   b) Have same name, different inputs
   c) Have same name, same inputs
   d) Not exist in C++

Exercise 4: Overload It! (Coding Choice)

Write 2 versions of `sayHi`:

```
void sayHi() {
    std::cout << "Hi!";
}

void sayHi(__________) {         // [1]
    std::cout << "Hi, " << name << "!";   // [2]
}
```

Then in `main()`:

```
sayHi();              // [3]
sayHi("Zimo");        // [4]
```

✅ Answers: Chapter 7

Exercise 1

1. c
2. d
3. c

4. a
5. c

Exercise 2

1. `b`
2. `multiply`
3. already correct

Exercise 3

1. c
2. b
3. c
4. c
5. b

Exercise 4

1. `std::string name`
2. already correct
3. already correct
4. already correct

📘 C++ for Clever Kids – Exercise Book

✨ *Chapter 8: Let's Build Something Big (Classes & Objects)*

Exercise 1: Class Basics (Single Choice)

1. What is a class?
   a) A kind of number
   b) A blueprint for objects
   c) A function
   d) A loop

2. What is an object?
   a) A loop in a function
   b) A copy of a class
   c) A comment
   d) A pointer

3. How do you make a class public?
   a) Use `open:`
   b) Use `shared:`
   c) Use `public:`
   d) Use `show:`

4. What keyword runs when an object is created?
   a) function
   b) object
   c) constructor
   d) create

5. Which one is used to *clean up* after a class ends?
   a) `delete`
   b) `~MyClass()`
   c) `main()`
   d) `public void()`

Exercise 2: Complete the Class (Coding Fill-in-the-Blank)

```
class Animal {
  public:
    std::string name;
```

```
    void speak() {
        std::cout << "I am a ____!";   // [1]
    }
};

int main() {
    Animal dog;
    dog.____ = "Dog";                // [2]
    dog.____();                      // [3]
}
```

Exercise 3: What Will It Print? (Single Choice)

1.

```
class Box {
  public:
    int value = 10;
};

Box b;
std::cout << b.value;
```

a) value
b) 10
c) Error
d) Box

2.

```
class Person {
  public:
    void sayHi() {
        std::cout << "Hi!";
    }
};
```

How do you call the function?

a) `Person.sayHi();`
b) `sayHi();`
c) `myPerson.sayHi();`
d) `Person::sayHi();`

3. Which is a valid constructor?

    a) `MyClass()`
    b) `~MyClass()`
    c) `construct MyClass()`
    d) `start()`

4. What will this print?

```
class Car {
  public:
    std::string brand = "Zoomy";
};

Car c;
std::cout << c.brand;
```

a) brand
b) Zoomy
c) Car
d) Error

5. What happens when you try to access a private variable outside the class?
    a) It works
    b) You get a warning
    c) You get an error
    d) It becomes public

Exercise 4: Get & Set (Coding Choice)

```
class Player {
  private:
    int score;

  public:
    void setScore(int s) {
        ________ = s;         // [1]
    }

    int getScore() {
        return ________;      // [2]
    }
};

int main() {
    Player p;
    p.setScore(100);
    std::cout << p.getScore();   // [3]
}
```

✅ Answers: Chapter 8

Exercise 1

1. b
2. b
3. c
4. c
5. b

Exercise 2

1. animal

2. name
3. speak

Exercise 3

1. b
2. c
3. a
4. b
5. c

Exercise 4

1. `score`
2. `score`
3. Already correct

📘 C++ for Clever Kids – Exercise Book

✨ *Chapter 9: The Secret Wizard (Preprocessor)*

Exercise 1: What's That Symbol? (Single Choice)

1. What does every preprocessor line start with?
   a) `/`
   b) `!`
   c) `#`
   d) `@`

2. What does `#include <iostream>` do?
   a) Adds numbers
   b) Shows text on screen
   c) Adds a library to use
   d) Deletes old code

3. What does `#define PI 3.14` do?
   a) Makes a function
   b) Creates a variable
   c) Makes a shortcut
   d) Does nothing

4. What is the purpose of `#ifdef DEBUG`?
   a) Always runs code
   b) Runs code only if DEBUG is defined
   c) Makes a comment
   d) Stops code

5. What is `#endif` used for?
   a) Ends the whole program
   b) Ends a function
   c) Ends a loop
   d) Ends a conditional preprocessor block

Exercise 2: Fill the Gaps (Coding Fill-in-the-Blank)

```
// Add a line to include the iostream library
________ <iostream>       // [1]

// Define a constant value
#define SPEED ________     // [2]

// Use a conditional block
#ifdef ________           // [3]
    std::cout << "Debug Mode";  // [4]
#endif                    // [5]
```

Exercise 3: What Will It Print or Do? (Single Choice)

1.

```
#define HELLO "Hi"
std::cout << HELLO;
```

a) HELLO
b) Hi
c) Error
d) ""

2.

```
#define X 5
#define Y 10
std::cout << X + Y;
```

a) 15
b) X + Y
c) 510
d) 5 + 10

3. What happens if `#define DEBUG` is not written and this is used:

```
#ifdef DEBUG
std::cout << "Debug!";
#endif
```

a) It prints Debug!
b) Error
c) Nothing happens
d) Debug is printed twice

4. Which line causes a compiler error if the condition is true?
   a) `#ifdef DEBUG`
   b) `#error "Stop here!"`

c) `#define PI 3.14`
d) `#include <math.h>`

5. What does `__FILE__` represent?
   a) The number of lines
   b) The filename
   c) A list of functions
   d) The last error

Exercise 4: Macro Magic (Coding Choice)

```
// Define a macro that doubles a number
#define DOUBLE(x) ________    // [1]

// Use it to print double of 4
std::cout << DOUBLE(4);       // [2]

// Use it to double a variable
int n = 5;
std::cout << DOUBLE(n);       // [3]

// Create a string from a word using #
#define STR(x) ________       // [4]

std::string msg = STR(Hello); // [5]
```

✅ Answers: Chapter 9

Exercise 1

1. c
2. c
3. c

4. b
5. d

Exercise 2

1. `#include`
2. `100` (or any number)
3. `DEBUG`
4. already correct
5. already correct

Exercise 3

1. b
2. a
3. c
4. b
5. b

Exercise 4

1. `((x) * 2)`
2. `8`
3. `10`
4. `#x`
5. `"Hello"`

📘 C++ for Clever Kids – Exercise Book

✨ *Chapter 10: Quick Extras (Little Big Things!)*

Exercise 1: Escape It! (Single Choice)

1. What does `\n` do in a string?
   a) Adds a tab
   b) Goes to a new line
   c) Adds a slash
   d) Ends the program

2. What does `\\` print?
   a) Two slashes
   b) Nothing
   c) One slash
   d) Double quotes

3. Which escape sequence adds a tab space?
   a) `\r`
   b) `\0`
   c) `\t`
   d) `\f`

What will this print?

```
std::cout << "Hi\nBye";
```

4. a) Hi Bye
   b) Hi
   Bye
   c) Hi\nBye
   d) Error

5. Which of these prints a double quote?
   a) `std::cout << "\'";`
   b) `std::cout << "\"";`

c) `std::cout << "/"";`
d) `std::cout << "'";`

Exercise 2: Keyword Detective (Multiple Choice)

1. Which of these are keywords in C++?
   a) `int`
   b) `return`
   c) `myVar`
   d) `if`

2. What kind of word is `const`?
   a) A comment
   b) A keyword
   c) A function
   d) A variable

3. What does `continue` do in a loop?
   a) Ends the program
   b) Jumps to the next loop step
   c) Returns a value
   d) Nothing

4. What does `bool` mean?
   a) True or false
   b) A number
   c) A loop
   d) A condition

5. What can `class` be used for?
   a) Declaring a new loop
   b) Defining a new type of object
   c) Writing a return function
   d) Making an error

Exercise 3: Code Matcher (Single Choice)

Match each symbol to what it does.

1. `==`
   a) Makes a value
   b) Compares two things
   c) Starts a loop
   d) Ends a string

2. `=`
   a) Adds numbers
   b) Checks if equal
   c) Assigns a value
   d) Ends a loop

3. `!`
   a) Opposite
   b) Loop
   c) Join
   d) Comment

4. `&&`
   a) Or
   b) If
   c) And
   d) Not

5. `||`
   a) Or
   b) End
   c) Add
   d) New line

Exercise 4: Mixed Magic (Coding Choice)

```
// Use escape sequences
std::cout << "Line1\nLine2";          // [1]

// Print a quote
std::cout << "She said: \"Hi!\"";     // [2]
```

```
// Use a const variable
const int year = ____;                  // [3]

// Use the keyword for a floating-point number
______ price = 4.99;                    // [4]

// Create a boolean variable
bool isHappy = ______;                 // [5]
```

✅ Answers: Chapter 10

Exercise 1

1. b
2. c
3. c
4. b
5. b

Exercise 2

1. a, b, d
2. b
3. b
4. a
5. b

Exercise 3

1. b

2. c
3. a
4. c
5. a

Exercise 4

1. already correct
2. already correct
3. `2025` (or any year)
4. `float`
5. `true` or `false`

📘 C++ for Clever Kids – Final Challenge!

🎓 *One last test to show how much you've learned!*

Exercise 1: Code Shuffle

What does this print?

```
std::cout << "Hi\nBye";
```

1.

Fill in the blank:

```
int _____ = 10;
```

2.
3. What symbol checks for equality?
   a) =
   b) ==
   c) !=
   d) :=

4. Which loop runs at least once?
   a) while
   b) for
   c) do-while
   d) switch

5. What keyword is used to make your own object?
   a) function
   b) class
   c) define
   d) loop

Exercise 2: Mini Fixes

What's missing?

```
std::cout << "Hello!" << ____;
```

1.
2. Which of these is a valid string?
   a) 'Hi'
   b) "Hi"
   c) Hi
   d) \Hi\

3. What will `std::cin >> name;` do?

4. Can `const int lucky = 7;` ever change?

5. Write the keyword to **exit** `main()` with success.

Exercise 3: Thinking in Loops

1. What does `i++` do?

Fill in:

```
for (int i = 0; i < 3; i++) {
    std::cout << i;
}
```

2.
3. What does `break;` do in a loop?

4. Which loop continues forever?

What's printed:

```
for (char c : std::string("Yo")) std::cout << c << " ";
```

5.

Exercise 4: Quick Pick

1. `char` can store:

2. What does `#include` do?

3. Which of these is a valid function header?

4. What's `STR(x)` in a macro?

5. What is `__LINE__`?

Exercise 5: Arrays & Friends

1. What does `marks[0]` access?

2. What happens if you go outside array size?

Fill:

```
std::array<int, 3> nums = {1, 2, ____};
```

3.
4. How do you print the second row, third column of a 2D array?

5. What's the last index of a 5-element array?

Exercise 6: Say My Name

What's the result of:

```
void greet(std::string name) {
    std::cout << "Hi, " << name;
}
```

1.
2. Can functions have the same name with different inputs?

What's the return type of:

```
int add(int a, int b);
```

3.

Fill in:

```
return ____;
```

4.
5. Which type returns nothing?

Exercise 7: If You Say So

What does this do?

```
if (age > 10)
```

1.
2. What happens if condition is false?
3. What operator is used for “or”?
4. What operator is used for “not equal”?

Fill:

```
(score >= 90) ? "Great" : "Keep trying";
```

5.

Exercise 8: Class Power

What’s this?

```
class Book { };
```

1.
2. What does `~Book()` mean?
3. How do you make a class function public?
4. What keyword prevents changing a value?

Fill:

```
myObj.myMethod();
```

5.

Exercise 9: Wizard's Tricks

1. What does `#define MAX 10` do?

2. How do you stop the compiler with a message?

What's the output of:

```
std::cout << __FILE__;
```

3.

Use `#ifdef` to wrap:

```
std::cout << "Debug!";
```

4.
5. What does `#include <cmath>` allow?

Exercise 10: Mixed Magic

Fill:

```
bool isOn = _____;
```

1.
2. What prints a double quote?

3. Which keyword makes a function stop and return something?

4. What does `auto` do?

What's printed:

```
std::cout << "A\\B";
```

5.

Exercise 1

1.

```
Hi
Bye
```

2. `int number = 10;`
3. b) `==`
4. c) `do-while`
5. b) `class`

Exercise 2

1. `std::endl`
2. b) `"Hi"`
3. It waits for the user to type input into `name`
4. No, it's constant
5. `return 0;`

Exercise 3

1. Adds 1 to `i`
2. Already filled

3. Ends the loop immediately
4. `while (true)`
5. `Y o`

Exercise 4

1. A single character
2. Adds a header/library
3. `int fun()` or `void fun()`
4. Turns name into `"name"`
5. Shows the current line number

Exercise 5

1. The first element
2. Undefined behavior / error
3. `3`
4. `array[1][2]`
5. Index `4`

Exercise 6

1. Prints: `Hi, <name>`
2. Yes, that's function overloading
3. `int`
4. A value (e.g. `x`)
5. `void`

Exercise 7

1. Checks if age is over 10
2. The `else` block runs (if present), or nothing
3. `||`
4. `!=`
5. Ternary operator for conditionals

Exercise 8

1. Defines a class
2. Destructor (runs when object is deleted)
3. `public:`
4. `const`
5. Calls a method on the object

Exercise 9

1. Makes a constant shortcut
2. `#error "Your message"`
3. The name of the file (e.g. `"main.cpp"`)

```
#ifdef DEBUG
std::cout << "Debug!";
#endif
```

4.
5. Gives access to math functions like `sqrt()`

Exercise 10

1. `true` or `false`
2. `\"`
3. `return`
4. Automatically guesses the variable type
5. `A\B`