DOCUMENTATION

Paolo Navatta

1
About CLEER

What is CLEER?
CLEER is a general-purpose, object-oriented programming language based on C++, originally created in 2023 by Paolo Navatta.

Why CLEER?
CLEER offers a syntax closely resembling C++, but designed to be more intuitive and straightforward.

What does CLEER mean?
The name CLEER was born as an assonance of the word "Clear", highlighting the programming language's clarity. However, it can also be seen as a construct made using the letter C (since the language is entirely based on C) and LEER (which is the Spanish word for "read"): a readable C.

2

Variables

```
int num = 1;
/* stores integer numbers */

short smallNum = 100;
/* stores short integer numbers */

long long largeNum = 1000000;
/* stores large integer numbers */

float floatNum = 3.14;
/* stores floating point numbers */

double doubleNum = 3.14159;
/* stores double precision floating point numbers */

char letter = 'A';
/* stores single characters */

string text = "Hello";
/* stores text as a string */

bool stat = true;
/* stores boolean values (true or false) */

int array[5];
/* stores an array of integers */

int* ptr = nullptr;
/* stores the address of an integer variable (pointer) */

var auto = "CLEER";
/* stores any variable type */
```

3

Functions

```
int myInt() {
    /* function that returns an integer */
    return 0;
}

float myFloat() {
    /* function that returns a float */
    return 3.1;
}

double myDouble( /* parameters (optional) */ ) {
    /* function that returns a double */
    return 2.71828;
}

char myChar( /* parameters (optional) */ ) {
    /* function that returns a character */
    return 'A';
}

string myString( /* parameters (optional) */ ) {
    /* function that returns a string */
    return "Hello";
}

bool myBool( /* parameters (optional) */ ) {
    /* function that returns a boolean */
    return true;
}
```

```
void myVoid( /* parameters (optional) */ ) {
    /* function that does not return anything */
}

func myFunc( /* parameters (optional) */ ) {
    /* function that can return any value*/
    return 0;
}
```

MAIN

CLEER's entry point is the `main` function, where you write your code and invoke other functions.

```
main {
    myFunc();
}
```

4

Comments

```
/* Comment */
```

5

Statements

IF

```
if (x == y) {
    /* code executed if x equals y */
} else {
    /* code executed if x doesn't equal y */
}
```

FOR

```
for (int i = 0; i < 10; i++) {
    /* code executed inside the loop */
}
```

WHILE

```
while (i < 10) {
    /* code executed inside the loop */
    i++;
}
```

DO-WHILE

```
do {
    /* code executed inside the loop */
    i++;
} while (i < 10);
```

BREAK

```
break;     /* exit the loop */
```

CONTINUE

```
continue;     /* skips the rest of the
current loop iteration */
```

6

Operators

```
/* Arithmetic */
1 + 1   /* addition */
1 - 1   /* subtraction */
1 * 1   /* multiplication */
1 / 1   /* division */
1 % 1   /* modulus (remainder) */

/* Assignment */
x = 5   /* assignment */
x += 3  /* addition assignment */
x -= 2  /* subtraction assignment */
x *= 4  /* multiplication assignment */
x /= 2  /* division assignment */
x %= 3  /* modulus assignment */

/* Comparison */
1 == 1  /* equal to */
1 != 2  /* not equal to */
1 < 2   /* less than */
1 > 2   /* greater than */
1 <= 2  /* less than or equal to */
1 >= 2  /* greater than or equal to */

/* Logical */
true && false   /* logical AND */
true || false   /* logical OR */
!true           /* logical NOT */


/* Increment/Decrement */
x++     /* post-increment */
x--     /* post-decrement */
```

```
++x     /* pre-increment */
--x    /* pre-decrement */

/* Bitwise */
1 & 1   /* bitwise AND */
1 | 1   /* bitwise OR */
1 ^ 1   /* bitwise XOR */
~1      /* bitwise complement (NOT) */
1 << 2  /* left shift */
1 >> 2  /* right shift */
```

7

Error Handling

```
try {
    /* block of code where an exception may
occur */

} catch (error) {
    /* Block of code to handle the exception */

}
```

8

Input

```
console.input(word);
 /* ask for a character input */
```

9

Output

```
console.print("Hello World!");
/* print “Hello World” */


console.println("Hello World");
 /* print “Hello World” in a new line */
```

10

Strings

```
string text = "Hello World!";

string get = text.get(0, 3);
 /* get a string character or characters */

int size = text.size();  /* get string size */

text.erase(0, 3);
/* erase string characters */

string append = text.append(" This is
CLEER!");
 /* append a string to another */

int compare = text.compare(compare);
 /* compare 2 strings */

bool empty = text.empty();
/* returns true if the string is empty */
```

12

Arrays

```
string cars[4] = {"Volvo", "BMW", "Ford",
"Mazda"};  /* Define an array called cars with 4
elements  */

string volvo = cars[0];   /* Get the first element of
cars (Volvo)  */


int size = cars.size();    /* Get the size of cars (4) */
```

13

Pointers

```
string helloworld = "Hello World";

string* ptr = &helloworld;  /* A pointer variable, with the name ptr, that stores the address of helloworld. */

string value = *ptr;   /* A variable with the value of helloworld (Hello World) */
```

14

Libraries

```
#include<library>
 /* Include the library "library" in your project */
```

14.5

Libraries:
OS

```
#include <os>
 /* Include the library os in your project */

os.move("file.txt","files/file.txt");
 /* Move the file.txt in the folder files */

os.overwrite("file.txt","Hello World!");
 /* Overwrite the full content of file.txt */

os.write("file.txt","Hello World!");
 /* Write at the end of file.txt */

string content = os.read("file.txt");
 /* Get the content of file.txt */

os.create("file.txt");
 /* Create a file called “file.txt“ */

int content = os.size("file.txt");
 /* Get the byte-size of file.txt */
```

```
os.ls("/home");
 /* List all the files in the path /home */

string path = os.pwd();
 /* Get the current working directory */

os.start("image.png");
 /* Open with xdg-open image.png
(NOTE: You may need to install xdg-utils)  */

os.command("cleer file.cleer");
 /* Execute the command “cleer file.cleer” */
```