**CODE ASSIGNMENT UNIT 2**

**26. Hello World**

cpp

CopyEdit

#include <iostream>

int main() {

std::cout << "Hello, World!" << std::endl;

return 0;

}

**27. Sum of Two Command Line Integers**

cpp

CopyEdit

#include <iostream>

#include <cstdlib>

int main(int argc, char\* argv[]) {

if (argc != 3) {

std::cout << "Usage: ./program <num1> <num2>" << std::endl;

return 1;

}

int sum = std::atoi(argv[1]) + std::atoi(argv[2]);

std::cout << "Sum: " << sum << std::endl;

return 0;

}

**28. Factorial Function Prototype**

cpp

CopyEdit

int factorial(int n);

**29. Max of Three Numbers**

cpp

CopyEdit

#include <iostream>

int max(int a, int b, int c);

int main() {

std::cout << "Max: " << max(10, 20, 15) << std::endl;

}

int max(int a, int b, int c) {

return (a > b && a > c) ? a : (b > c ? b : c);

}

**30. Area of Circle**

cpp

CopyEdit

#include <iostream>

#define PI 3.14159

double area(double radius) {

return PI \* radius \* radius;

}

**31. Product of Two Integers**

cpp

CopyEdit

int product(int a, int b) {

return a \* b;

}

**32. Print Array Elements**

cpp

CopyEdit

#include <iostream>

void printArray(int arr[], int size) {

for (int i = 0; i < size; ++i)

std::cout << arr[i] << " ";

std::cout << std::endl;

}

**33. Swap Using Call by Reference**

cpp

CopyEdit

void swap(int &a, int &b) {

int temp = a;

a = b;

b = temp;

}

**34. Increment by 10 Using Reference**

cpp

CopyEdit

void incrementByTen(int &x) {

x += 10;

}

**35. Modify Array Using Reference**

cpp

CopyEdit

void modifyArray(int (&arr)[5]) {

for (int i = 0; i < 5; ++i)

arr[i] += 1;

}

**36. Inline Function to Square a Number**

cpp

CopyEdit

inline int square(int x) {

return x \* x;

}

**37. Inline Function to Cube a Number**

cpp

CopyEdit

inline int cube(int x) {

return x \* x \* x;

}

**38. Sum Using Inline Function**

cpp

CopyEdit

#include <iostream>

inline int sum(int a, int b) {

return a + b;

}

int main() {

std::cout << "Sum: " << sum(4, 5) << std::endl;

}

**39. Macro vs Inline for Square**

cpp

CopyEdit

#define SQUARE(x) ((x) \* (x))

inline int squareFunc(int x) { return x \* x; }

**40. Macro vs Inline for Max**

cpp

CopyEdit

#define MAX(x, y) ((x) > (y) ? (x) : (y))

inline int maxFunc(int a, int b) { return (a > b) ? a : b; }

**41. Overloaded Area Functions**

cpp

CopyEdit

double area(double radius) { return 3.14159 \* radius \* radius; }

int area(int length, int width) { return length \* width; }

double area(double base, double height, bool triangle) { return 0.5 \* base \* height; }

**42. Overloaded Max Functions**

cpp

CopyEdit

int max(int a, int b) { return (a > b) ? a : b; }

int max(int a, int b, int c) { return max(max(a, b), c); }

**43. Overloaded Print Functions**

cpp

CopyEdit

void print(int i) { std::cout << i << std::endl; }

void print(float f) { std::cout << f << std::endl; }

void print(std::string s) { std::cout << s << std::endl; }

**44. Compound Interest with Default Args**

cpp

CopyEdit

double compoundInterest(double p, double r = 5.0, int t = 2) {

return p \* pow((1 + r / 100), t);

}

**45. Greeting with Default Name**

cpp

CopyEdit

void greet(std::string name = "Guest") {

std::cout << "Hello, " << name << "!" << std::endl;

}

**46. Power Function with Default Exponent**

cpp

CopyEdit

double power(double base, int exponent = 2) {

return pow(base, exponent);

}

**47. Recursive Function Demo (Factorial)**

cpp

CopyEdit

int factorial(int n) {

if (n <= 1) return 1;

return n \* factorial(n - 1);

}

**48. Array of Function Pointers**

cpp

CopyEdit

#include <iostream>

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

int multiply(int a, int b) { return a \* b; }

int main() {

int (\*funcPtr[2])(int, int) = { add, multiply };

std::cout << "Add: " << funcPtr[0](2, 3) << std::endl;

std::cout << "Multiply: " << funcPtr[1](2, 3) << std::endl;

}

**49. Function Template Example**

**Description**: A function template allows the same function to operate on different data types.

cpp

CopyEdit

#include <iostream>

using namespace std;

// Template for finding the maximum of two values

template <typename T>

T maximum(T a, T b) {

return (a > b) ? a : b;

}

int main() {

cout << "Max of 3 and 7: " << maximum(3, 7) << endl;

cout << "Max of 4.5 and 2.3: " << maximum(4.5, 2.3) << endl;

cout << "Max of 'A' and 'Z': " << maximum('A', 'Z') << endl;

return 0;

}

**50. Function Pointers and Callback Functions**

**Description**: A function pointer stores the address of a function and can be used for callbacks.

cpp

CopyEdit

#include <iostream>

using namespace std;

// A callback function

void greet() {

cout << "Hello from callback!" << endl;

}

// Another callback

void farewell() {

cout << "Goodbye from callback!" << endl;

}

// Function that takes a function pointer (callback)

void performAction(void (\*callbackFunc)()) {

cout << "Performing action..." << endl;

callbackFunc(); // Call the passed function

}

int main() {

performAction(greet);

performAction(farewell);

return 0;

}