**Contents**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | 1 Introduction to C |  | | 1.1 The components of a C program | 1 | | 1.2 DECLARE VARIABLES AND ASSIGN VALUES | 3 | | 1.3 Properties of function and C keywords | 5 | | 1.4 USE FUNCTIONS TO RETURN VALUES | 6 | | 1.5 Arguments of functions and their use | 7 | | 1.6 REMEMBER THE C KEYWORDS | 7 | |  |  | | 2 C Control Statements |  | | 2.1 The Block of statements or Code Bolcks | 8 | | 2.2 The Selection statements | 8 | | 2.3 The "if" statement | 8 | | 2.4 The "if-else" statement | 8 | | 2.5 The "for" loop | 9 | | 2.6 The increment operator "++" and decrement operator " - - " | 9 | | 2.7 The backslash charecters \n "newline", \t "tab" etc | 10 | | 2.8 RELATIONAL AND LOGICAL OPERATORS | 10 | | NOTE : Execution of printf with ++ operators | 11 | | 2.9 Character inputs from keyboard | 11 | | 2.10 NEST if statements and if-else-if LADDER | 11 | | 2.11 for Loop – 'Advanced' | 13 | | 2.12 while Loop | 13 | | 2.13 do Loop | 14 | | 2.14 NESTED LOOPS | 14 | | 2.15 Use break TO EXIT A LOOP | 15 | | 2.16 Use continue for skipping any code | 15 | | 2.17 The switch STATEMENT | 16 | | 2.18 The goto STATEMENT | 16 | | 2.19 C control statements | 17 | |  |  | | 3 Advanced data types, variables & Expression |  | | 3.1 DATA - TYPE MODIFIERS | 18 | | 3.2 Advanced variable declaration (Local and Global) | 19 | | 3.3 Constants Advanced | 20 | | 3.4 Variable initialization | 21 | | 3.5 Type conversion in "Expression" and "Assignment" | 21 | | 3.6 The TYPE CASTS | 22 | |  |  | | 4 Strings, Arrays & Pointers |  | | 4.1 One dimensional Arrays | 23 | | 4.2 USE STRINGS : gets(), 4-string functions, atoi(), STRING.H & STDLIB.H | 24 | | 4.3 Create multidimensional Arrays | 25 | | 4.4 Initialize Arrays | 25 | | 4.5 ARRAYS OF STRINGS | 26 | | 4.6 The POINTERS | 26 | | 4.7 Restriction to Pointer Expression | 27 | | 4.8 POINTERS WITH ARRAYS | 28 | | 4.9 Use pointers to string constants | 29 | | 4.10 Arrays of Pointers | 29 | | 4.11 Multiple INDIRECTION | 30 | | 4.12 Pointers as Parameters | 30 | |  |  | | 5 Functions in C |  | | 5.1 The PROTOTYPE | 31 | | 5.2 Recursion | 33 | | 5.3 Parameters Advanced | 34 | | 5.4 Pass Arguments To Main() | 36 | | 5.5 Old-Style Parameter Declarations | 37 | | 5.6 variable storage class specifire (Advanced topic) | 37 | | 5.7 Access modifiers : const and volatile (Advanced topic) | 39 | | 5.8 Function Pointers (Advanced topic) | 40 | |  |  | | 6 C's Console input/output & File input/output |  | | 6.1 What is console? With brief intro : | 41 | | 6.2 Macro substitution : #define | 41 | | 6.3 Standard CONSOLE i/o : getchar(), putchar() & EOF | 42 | | 6.4 NON-STANDARD CONSOLE FUNCTIONS : getche(), getch(), kbhit(), cprintf(), cscanf() | 43 | | 6.5 Details on gets() and puts( ) | 43 | | |  |  | | --- | --- | | 6.6 printf( ) : Details | 43 | | 6.7 scanf( ) : Details | 45 | | 6.8 STREAMS for file I/O in C | 46 | | 6.9 File access using fopen(), fclose() and read/write using fgetc(), fputc() | 47 | | 6.10 End of file [EOF] feof() And file error checking ferror() | 50 | | 6.11 String I/O in a File with fputs() & fgets(). Text I/O with fprintf() & fscanf() | 50 | | 6.12 READ AND WRITE BINARY DATA | 51 | | 6.13 Random access using fseek() | 52 | | 6.14 Some other important File-System functions | 53 | | 6.15 THE STANDARD STREAMS | 53 | |  |  | | 7 Custom (User-defined) data-types & Advanced  Operators | | | 7.1 The Custom (User-defined) Data-types of C | 54 | | 7.2 STRUCTURE Basics | 54 | | 7.2.1 Defining structures | 54 | | 7.2.2 Declaring structure variables : | 55 | | 7.2.3 Structure variable initialization : | 56 | | 7.2.4 Accessing members of a structure & use of "." operator | 56 | | 7.2.5 Structures as arrays : | 57 | | 7.2.6 Arrays within Structures : | 57 | | 7.2.7 COPYING AND COMPARING STRUCTURE VARIABLES | 57 | | 7.2.8 Structures and Functions | 57 | | 7.2.9 SIZE OF STRUCTURES | 59 | | 7.2.10 Declare Pointer to Structure | 59 | | 7.2.11 NESTED STRUCTURES | 60 | | 7.3 BIT FIELDS | 60 | | 7.4 UNIONS | 62 | | 7.4.1 Difference between structure and union | 62 | | 7.4.2 Assigning values to union members | 63 | | 7.5 ENUMERATIONS | 63 | | 7.6 typedef | 63 | | 7.7 Bitwise and Shift Operators | 64 | | 7.8 OPERATORs Advanced | 65 | | 7.8.1 The ternary operator " ? : " | 65 | | 7.8.2 The Comma Operator : | 65 | | 7.8.3 More Uses Of Assignment Operator | 65 | | 7.8.4 The precedence of all C - OPERATORS | 65 | |  |  | | 8 C Preprocessors and Advanced topics |  | | 8.1 Advanced #define and #include | 66 | | 8.2 Conditional COMPILATlON | 66 | | 8.3 #error, #undef, #Iine, #pragma | 67 | | 8.4 C's built-in MACROS | 68 | | 8.5 The # and ## operators | 68 | | 8.6 DYNAMIC ALLOCATION | 68 | |  |  | | 9 Introduction to C++ | | | 9.1 Get familiar with OOP (Object Oriented Programming) | 69 | | 9.2 Old Header And Standard Header Declaration Of C++ | 70 | | 9.3 Function overloading in C++ | 70 | | 9.4 C++ comments | 71 | | 9.5 C++ Console I/O | 71 | | 9.6 Difference between C and C++ | 72 | | 9.7 C++ Keywords | 72 | |  |  | | 10 C++ Class, Objects with array and pointer | | | 10.1 Introduction to CLASS | 73 | | 10.2 CONSTRUCTOR and DESTRUCTOR Functions | 74 | | 10.3 Constructors with Parameters | 74 | | 10.4 Relation between STRUCTURES-UNIONS and CLASSES | 75 | | 10.5 In-Line Functions & Automatic In-Lining | 76 | | 10.6 Assigning Objects | 76 | | 10.7 Object Pointers | 77 | | 10.8 The "this" pointer | 77 | | 10.9 ARRAYS OF OBJECTS | 78 | | 10.10 PASSING objects to functions and RETURNING objects from function | 79 | | 10.11 Memory allocation/release operators : new, delete | 81 | | 10.12 References | 82 | | |  |  | | --- | --- | | 11 Overloading: function & Operators. Inheritance. | | | 11.1 Introduction to INHERITANCE | 85 | | 11.2 Intro to FRIEND functions | 85 | | 11.3 Overloading CONSTRUCTOR | 87 | | 11.4 COPY CONSTRUCTOR (recall 10.10) | 88 | | 11.5 Default arguments | 89 | | 11.6 Ambiguity Caused By Overloading | 89 | | 11.7 Address of an OVERLOADED function (recall 5.8) | 90 | | 11.8 Overloading MEMBER OPERATOR FUNCTIONS | 90 | | 11.9 Overloading Binary Operators | 91 | | 11.10 Overloading the RELATIONAL and LOGICAL operators | 92 | | 11.11 Overloading A UNARY Operator | 93 | | 11.12 Overloading FRIEND OPERATOR FUNCTIONS | 93 | | 11.13 Assignment Operator Advanced | 94 | | 11.14 Overloading The [ ] SUBSCRIPT Operator | 95 | | 11.15 INHERITANCE: access control of base class | 96 | | 11.16 Accessing PROTECTED members | 97 | | 11.17 INHERITANCE with Constructors-Destructors | 97 | | 11.18 MULTIPLE INHERITANCE | 99 | | 11.19 VIRTUAL BASE (problems with "one derived" & "multiple direct base") | 100 | |  |  | | 12 C++ I/O system |  | | 12.1 C++ I/O Stream | 101 | | 12.2 Formatted I/O | 101 | | 12.3 width(), precision(), AND fill() | 103 | | 12.4 I/O MANIPULATORS | 103 | | 12.5 Inserters and Extractors | 104 | | 12.6 User Defined Manipulators | 105 | | 12.7 File I/O | 106 | | 12.8 UNFORMATTED I/O & BINARY I/O | 107 | | 12.9 Checking I/O Status | 109 | | 12.10 Random Access | 110 | | 12.11 Customized I/O And Files | 110 | |  |  | | 13 Polymorphism, Exceptions, RTTI, Operator cast |  | | 13.1 Pointers To Derived Classes | 111 | | 13.2 Virtual Functions (VF) | 111 | | 13.3 Abstract class and Pure Virtual function (PVF) | 113 | | 13.4 Polymorphism: Early binding & Late binding | 114 | | 13. 5 Generic-Functions & Generic-Classes (GnF & GnC) | 116 | | 13.6 EXCEPTION HANDLING | 118 | | 13.7 Handling exceptions thrown by new | 121 | | 13.8 RTTI (run-time type identification) | 122 | | 13.9 C++ casting operators | 124 | | 13.9.1 dynamic\_cast: | 124 | | 13.9.2 const\_cast, reinterpret\_cast and static\_cast | 125 | |  |  | | 14 Miscellaneous topics & STL |  | | 14.1 namespace Details | 127 | | 14.2 Conversion function (CvF) | 129 | | 14.3 static Class Members | 129 | | 14.4 const MEMBER FUNCTIONS AND mutable | 130 | | 14.5 Initializing object using "=" and the "explicit" specifier | 131 | | 14.6 LINKAGE specifier for linking other language. asm for linking assembly language. | 132 | | 14.7 ARRAY-BASED I/O (Not will be used) | 132 | | ***Standard Template Library (STL)*** | 133 | | 14.8 An Overview Of STL | 133 | | 14.9 Type Names (Placeholder Types) For Container Classes | 134 | | 14.10 VECTORS | 135 | | 14.11 LISTS | 137 | | 14.12 MAPS (example of an associative container) | 139 | | 14.13 ALGORITHMS (names of the algorithms with purpose) | 141 | | 14.14 STRING class | 143 | |

Appendix