1. Difference between arrays and pointers?

2. What is the purpose of realloc( )?

3. What is static memory allocation and dynamic memory allocation?

4. How are pointer variables initialized?

5. Are pointers integers?

6. What is a pointer variable?

7. What is a pointer value and address?

8. What is a method?

9. What are the advantages of the functions?

10. What is the purpose of main( ) function?

11. What is an argument ? differentiate between formal arguments and actual arguments?

12. What is a function and built-in function?

13. What is modular programming?

14. When does the compiler not implicitly generate the address of the first element of an array?

15. What are the characteristics of arrays in C?

16. Differentiate between a linker and linkage?

17. What are advantages and disadvantages of external storage class?

18. Diffenentiate between an internal static and external static variable?

19. What are the advantages of auto variables?

20. What is storage class and what are storage variable ?

21. Which expression always return true? Which always return false?

22. Write the equivalent expression for x%8?

23. why n++ executes faster than n+1?

24. what is a modulus operator? What are the restrictions of a modulus operator?

25. What is the difference between a string and an array?

26. Is it better to use a pointer to navigate an array of values,or is it better to use a subscripted array name?

27. Can the sizeof operator be used to tell the size of an array passed to a function?

28. Is using exit() the same as using return?

29. Is it possible to execute code even after the program exits the main() function?

30. What is a static function?

31. Why should I prototype a function?

32. How do you print an address?

33. Can math operations be performed on a void pointer?

34. How can you determine the size of an allocated portion of memory?

35. What is a “null pointer assignment” error? What are bus errors, memory faults, and core dumps?

36. What is the difference between NULL and NUL?

37. What is the heap?

38. Can the size of an array be declared at runtime?

39. What is the stack?

40. When should a far pointer be used?

41. What is the difference between far and near?

42. Is it better to use malloc() or calloc()?

43. Why should we assign NULL to the elements (pointer) after freeing them?

44. When would you use a pointer to a function?

45. How do you use a pointer to a function?

46. Can you add pointers together? Why would you?

47. What does it mean when a pointer is used in an if statement?

48. Is NULL always defined as 0?

49. What is a void pointer?

50. What is a null pointer?

51. How many levels of pointers can you have?

52. What is indirection?

53. How do you print only part of a string?

54. How can I convert a string to a number?

55. How can I convert a number to a string?

56. What is the difference between a string copy (strcpy) and a memory copy (memcpy)? When should each be used?

57. How can you check to see whether a symbol is defined?

58. How do you override a defined macro?

59. What is #line used for?

60. What is a pragma?

61. What are the standard predefined macros?

62. How can type-insensitive macros be created?

63. How many levels deep can include files be nested?

64. Can include files be nested?

65. Can you define which header file to include at compile time?

66. What is the difference between #include and #include “file”?

67. Is it better to use a macro or a function?

68. How are portions of a program disabled in demo versions?

69. What is the benefit of using an enum rather than a #define constant?

70. What is the benefit of using #define to declare a constant?

71. Can a file other than a .h file be included with #include?

72. How can you avoid including a header more than once?

73. What will the preprocessor do for a program?

74. What is a macro, and how do you use it?

75. What is Preprocessor?

76. How can I make sure that my program is the only one accessing a file?

77. How can I open a file so that other programs can update it at the same time?

78. How do you determine whether to use a stream function or a low-level function?

79. What is the difference between text and binary modes?

80. How can you restore a redirected standard stream?

81. How do you redirect a standard stream?

82. How can I search for data in a linked list?

83. How can I sort a linked list?

84. What is hashing?

85. What is the quickest searching method to use?

86. What is the easiest searching method to use?

87. How can I sort things that are too large to bring into memory?

88. What is the quickest sorting method to use?

89. What is the easiest sorting method to use?

90. What is the benefit of using const for declaring constants?

91. Can static variables be declared in a header file?

92. What is the difference between declaring a variable and defining a variable?

93. Is it acceptable to declare/define a variable in a C header?

94. When should a type cast not be used?

95. When should a type cast be used?

96. How can you determine the maximum value that a numeric variable can hold?

97. How reliable are floating-point comparisons?

98. Can a variable be both const and volatile?

99. When should the volatile modifier be used?

100.When should the register modifier be used? Does it really help?

Unix shell scripting questions……….

1. How do you find out what’s your shell? - echo $SHELL

2. What’s the command to find out today’s date? - date

3. What’s the command to find out users on the system? - who

4. How do you find out the current directory you’re in? - pwd

5. How do you remove a file? - rm

6. How do you remove a < in files the all with>- rm -rf

7. How do you find out your own username? - whoami

8. How do you send a mail message to somebody? - mail somebody@techinterviews.com -s ‘Your subject’ -c ‘cc@techinterviews.com‘

9. How do you count words, lines and characters in a file? - wc

10. How do you search for a string inside a given file? - grep string filename

11. How do you search for a string inside a directory? - grep string *

12. How do you search for a string in a directory with the subdirectories recursed? - grep -r string *

13. What are PIDs? - They are process IDs given to processes. A PID can vary from 0 to 65535.

14. How do you list currently running process? - ps

15. How do you stop a process? - kill pid

16. How do you find out about all running processes? - ps -ag

17. How do you stop all the processes, except the shell window? - kill 0

18. How do you fire a process in the background? - ./process-name &

19. How do you refer to the arguments passed to a shell script? - $1, $2 and so on. $0 is your script name.

20. What's the conditional statement in shell scripting? - if {condition} then … fi

21. How do you do number comparison in shell scripts? - -eq, -ne, -lt, -le, -gt, -ge

22. How do you test for file properties in shell scripts? - -s filename tells you if the file is not empty, -f filename tells you whether the argument is a file, and not a directory, -d filename tests if the argument is a directory, and not a file, -w filename tests for writeability, -r filename tests for readability, -x filename tests for executability

23. How do you do Boolean logic operators in shell scripting? - ! tests for logical not, -a tests for logical and, and -o tests for logical or.

24. How do you find out the number of arguments passed to the shell script? - $#

25. What's a way to do multilevel if-else's in shell scripting? - if {condition} then {statement} elif {condition} {statement} fi

26. How do you write a for loop in shell? - for {variable name} in {list} do {statement} done

27. How do you write a while loop in shell? - while {condition} do {statement} done

28. How does a case statement look in shell scripts? - case {variable} in {possible-value-1}) {statement};; {possible-value-2}) {statement};; esac

29. How do you read keyboard input in shell scripts? - read {variable-name}

30. How do you define a function in a shell script? - function-name() { #some code here return }

31. How does getopts command work? - The parameters to your script can be passed as -n 15 -x 20. Inside the script, you can iterate through the getopts array as while getopts n:x option, and the variable $option contains the value of the entered option.