|  |  |  |
| --- | --- | --- |
| S NO | TITLE | Page No |
| 1 | What is an OS? | 3 |
| 2 | Write down the smallest executable code? | 4 |
| 3 | What are entry control and exit control loops? | 4 |
| 4 | What is the difference between including the header file with-in angular braces < > and double quotes ” “? | 4 |
| 5 | What is the difference between declaration and definition of a variable/function | 4 |
| 6 | What is C language? | 4 |
| 7 | Why C language known as the mother language? | 4 |
| 8 | Why is C called a mid-level language? | 4 |
| 9 | What are the features of C? | 4 |
| 10 | What is the use of scanf() and printf() function in C? | 5 |
| 11 | What are the difference between local nd global variable? | 5 |
| 12 | What is a storage class and What are different storage class specifiers in C? | 5 |
| 13 | What is a static variable? | 5 |
| 14 | Write a program that differentiates auto and static variables with the output? | 5 |
| 15 | What is the use of the function in c? | 6 |
| 16 | What is the difference between call by value and call by reference in C? | 6 |
| 17 | What is recursion in C? | 8 |
| 18 | What is an array in C? | 8 |
| 19 | What is a pointer in C? | 8 |
| 20 | What is the usage of a pointer in C? | 9 |
| 21 | What is a NULL pointer? | 9 |
| 22 | What is a Dangling pointer? | 9 |
| 23 | What is a pointer to a pointer? | 9 |
| 24 | Write a hello world program without using semicolon? | 10 |
| 25 | What is an infinite loop? | 10 |
| 26 | What is typecasting? | 10 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |
| --- |
| **1.What is an OS?**  Our computer may have many resources, we need 1 master who handles all these resources. This master is called is Operating System.  Ok! What all this master will handle?  Memory Management:  We may give different tasks to the computer. There should be a space for each of these processes. One task should not affect the space of others. This memory is managed by OS.  Process Management:  If we have 20 problems to solve, we cannot solve everything at a time, we assign time for each task. Similarly, the computer will also assign time for each task based on the round-robin algorithm.  Interprocess Communication:  Sometimes solution of question 1 can help for solution 10, same as that 1 process can be useful for another process. So we should link 2 processes. This task is done by OS.  File System:  OS manages files.  We have 2 modes in OS.  1.User mode:  It is the place where all our Basic programming will be stored. But when we are executing these codes we have to switch to Kernel mode(this switching is discussed below).  2.Kernal mode:  It is the place where all the above-discussed topics come under.  How to switch to kernel mode?  When we call print in user mode it will start the kernel-mode which will call the write function which will make the hardware work. In kernel mode, we have permission to access hardware. |

|  |
| --- |
| **2.Write down the smallest executable code?**  All code should have the main function. statements are not important.  void main()  {  } |
| **3.What are entry control and exit control loops?**  **entry control:**  will check the condition when we enter into the loop. (while, for)  **exit control:**  will check the condition at the end of the loop. (do-while). |
| **4.What is the difference between including the header file with-in angular braces < > and double quotes ” “?**  angular braces only check for the particular header file whereas quotes check for the relatable header files |
| **5.What is the difference between declaration and definition of a variable/function?**  Declaring a variable appears anywhere in the code in which no memory is allocated, but it will tell the compiler data type of the variable.  Defining a variable is a subset of declaring which allocates the memory for all the variables. |
| **6.What is C language?**  C is a mid-level, structured(procedural) language, here the problem is divided into submodules and each module is solved in a structured manner. This technique minimizes error and misinterpretation. |

|  |
| --- |
| **7.Why C language is known as the mother language?**  C is known as the mother language because most of the JVM’s and compilers are written in c and most of the languages are developed by C |
| **8.Why C is called as mid level language?**  C is a mid level language because it binds both high-level language and low-level language.we can create an OS as well as a menu driven program using C language. |
| **9.What are the features of C?**  Simple  Portable  mid-level language  Structured  Fast |
| **10.What is the use of scanf() and printf() function in C?**  scanf() is used to get the input from the user and printf() is used to display the value. |
| **11.What are the difference between local and global variable?**   |  |  |  | | --- | --- | --- | | **Comparison** | **Local** | **Global** | | Declaration | Variables are declared inside the function or a block | Variables are declared outside the function or a block | | Scope | Variables are available inside the scope in which they are declared | Variables are available in the whole program | | Access | Variables are accessed only inside the functions in which they are declared | Variables are accessed in the whole program | | Life | When the function/block gets terminated the variable also gets terminated | the variable also gets terminated  only when the program gets terminated | | Memory | memory is allocated in the stack unless specified | memory is allocated by the compiler. | |
| **12.What is a storage class and** **What are different storage class specifiers in C?**  A storage class specifies the visibility and the location of a variable declared.It describes where to declare a variable, what is its scope, who can(which function) can access the variable, what is its lifetime.  **different storage class specifiers**  auto, static, register, extern.  **Storage class mainly decides the scope of a variable:**  1. Auto - local  2. Register - within a block (eg if-else block or for loop, etc)  3. Static - within the same program /file  4. Extern - has a link some elsewhere in program |
| **13.What is a static variable?**  A static variable does not create a new one each time and does not destroy when the program’s execution leaves the function. It declares once and destroys once when the program’s execution finishes. |
| **14. Write a program that differentiates auto and static variables with the output?**  **#include <stdio.h>**  **void fun(void)**  **{**  **auto int a=0;**  **static int b=0;**    **printf("a = %d, b = %d\n",a,b);**    **a++;**  **b++;**  **}**  **int main()**  **{**  **int loop;**    **//calling function 10 times**  **for(loop=0; loop<5; loop++)**  **fun();**    **return 0;**  **}** |
| **15.What is the use of the function in c?**  Functions are used for the reusability of the code.  A program is divided into sub blocks so that we can check the flow of the code easily.  A function can be called any number of times. |
| **16.What is the difference between call by value and call by reference in C?**  Following are the differences between a call by value and call by reference are:   |  |  | | --- | --- | | **call by value** | **call by reference** | | A duplicate of original variables are passed | The address of the original value is passed | | the actual variable and the formal variable has a different memory location | the actual variable and the formal variable has the same memory location | | When there is any mistake in the value, the original value can be retrieved from the actual variable. | When there is any mistake in the value, the original value then we cannot retrieved it |   **Example of call by value:**   1. **#include <stdio.h>** 2. **void change(int,int);** 3. **int main()** 4. **{** 5. **int a=10,b=20;** 6. **change(a,b); //calling a function by passing the values of variables.** 7. **printf("Value of a is: %d",a);** 8. **printf("\n");** 9. **printf("Value of b is: %d",b);** 10. **return 0;** 11. **}** 12. **void change(int x,int y)** 13. **{** 14. **x=13;** 15. **y=17;** 16. **}**   **Output:**  **Value of a is: 10**  **Value of b is: 20**  **Example of call by reference:**   1. **#include <stdio.h>** 2. **void change(int\*,int\*);** 3. **int main()** 4. **{** 5. **int a=10,b=20;** 6. **change(&a,&b); // calling a function by passing references of variables.** 7. **printf("Value of a is: %d",a);** 8. **printf("\n");** 9. **printf("Value of b is: %d",b);** 10. **return 0;** 11. **}** 12. **void change(int \*x,int \*y)** 13. **{** 14. **\*x=13;** 15. **\*y=17;** 16. **}**   **Output:**  **Value of a is: 13**  **Value of b is: 17** |
| **17. What is recursion in C?**  When a function calls itself this process is known as recursion and the relevant function is known as a recursive function.\  There are 2 types of recursion they are  **1.Winding Recursion:**  When the function calls itself and terminates when the condition is reached then the recursion is called Winding Recursion.  **2.Unwinding Recursion:**  The unwinding phase starts when the condition is reached, and the control returns to the original call. |
| **18.What is an array in C?**  An array is a collection of variables of the same type that are referenced by a common name.  2 types of array  Single dimensional array  Multidimensional array |
| **19.What is a pointer in C?**  Whenever a variable is declared a memory is allocated for it, this memory has an address value, a variable which holds this value is known as a pointer variable. pointer makes the code optimized and makes the execution fast.   1. #include <stdio.h> 2. **int** main() 3. { 4. **int** \*p; //pointer of type integer. 5. **int** a=5; 6. p=&a; 7. printf("Address value of 'a' variable is %u",p); 8. **return** 0; 9. } |
| **20.What is the usage of a pointer in C?**  **Accessing Elements**:  used to traverse through an array  **Dynamic memory allocation:**  memory can be allocated and deallocated during the run time.  **Call by reference:**  used to pass the address value during the function call  **Data structures:**  Used to create data structures like LinkedList, trees, graphs. |
| **21.What is a NULL pointer?**  When a pointer does not point to any value but NULL then the pointer is called NULL pointer. If we want to make the pointer NULL then we have to make its value as 0. |
| **22.What is a Dangling pointer?**  When a memory of pointer is deleted by another pointer then the first pointer will stil point to the same memory location. This pointer is called dangling pointer.  to avoid this dangling pointer we can make the pointer point towards NULL.   1. #include<stdio.h> 2. **void** main() 3. { 4. **int** \*ptr = malloc(constant value); //allocating a memory space. 5. free(ptr); //ptr becomes a dangling pointer. 6. ptr=NULL; //Now, ptr is no longer a dangling pointer. 7. } |
| **23.What is a pointer to a pointer?**  When a pointer points to the address of another pointer then it is called as pointer to a pointer. |
| **24.Write a hello world program without using semicolon?**   1. **#include<stdio.h>** 2. **void main(){** 3. **if(printf("hello world")){} // It prints the ?hello world? on the screen.** 4. **}**   **#include <stdio.h>**  **int main()**  **{**  **while(printf("Hello World"))**  **{**  **return 0;**  **}**    **}** |
| **25.What is an infinite loop?**  When a loop runs an infinite number of times without an end condition then the loop is called as infinite loop.  **Infinite For Loop:**   1. **for**(;;){ 2. //code to be executed 3. }   **Infinite While Loop:**   1. **while**(1){ 2. //code to be executed 3. }   **Infinite Do-While Loop:**   1. **do**{ 2. //code to be executed 3. }**while**(1); |
| **26.What is typecasting?**  The process of converting one data type into another data type is known as typecasting.  Syntax:  (type\_name) expression; |

|  |
| --- |
| **27.What is a newline escape sequence?**  a newline escape sequence is represented by ‘\n’. It is used to add newline in output screen. |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |