**UNIVERSITY COLLEGE OF ENGINEERING (BIT CAMPUS),**

**TIRUCHIRAPPALLI-620 024**

**Second Internal Assessment Test**

**Department of CSE / IT**

**Subject Code :**EC8393 **Subject Name :** FUNDAMENTALS OF DATA STRUCTURES IN C

**Date&Duration :**17.09.2018& 1.30Hrs**Marks:50**

**Degree/Branch:** B.E/ECESec’A’ **Year/Semester:** II / III

**PART-A**

**Answer *All* Questions 7 x 2 = 14**

1. Define function and its advantages.

A function is a group of statements that are executed whenever the function is called to perform a specific designated task.

 Advantages.

1. Readability
2. Easily Managed
3. Modular Structure
4. Individually Tested
5. What is null pointer with example?

**Null pointer** is a special reserved value of a **pointer**. A **pointer** of any type has such a reserved value. Formally, each specific **pointer** type ( int \* , char \* etc.) has its own dedicated **null**-**pointer** value. Conceptually, when a **pointer** has that **null** value it is not pointing anywhere

1. What is pointer to pointer with simple example?

A **pointer** to a **pointer** is a form of multiple indirection, or a chain of **pointers**. Normally, a **pointer** contains the address of a variable. When we define a **pointer**to a **pointer**, the first **pointer** contains the address of the second **pointer**, which points to the location that contains the actual value as shown below.

1. What is nesting of function with syntax?

A **nested function** is a **function** that is completely contained within a parent **function**. Any **function** in a program file can include a **nested function**. The primary difference between **nested functions** and other types of **functions** is that they can access and modify variables that are defined in their parent**functions**.

1. Define recursive function.

A **recursive function** (DEF) is a **function** which either calls itself or is in a potential cycle of **function** calls. As the definition specifies, there are two types of **recursive functions**. Consider a **function** which calls itself: we call this type of **recursion**immediate **recursion**.

1. Explain passing parameters.

**parameter passing**. **parameter passing** The mechanism used to **pass parameters** to a procedure (subroutine) or function. The most common methods are to **pass** the value of the actual **parameter** (call by value), or to **pass** the address of the memory location where the actual **parameter** is stored (call by reference).

1. What is the difference between structure andunion?

|  |  |
| --- | --- |
| **Structure** | **Union** |
| Struct keyword is used to define a structure. | Union keyword is used to define a union. |
| Members do not share memory in a structure. | Members share the memory space in a union. |
| Any member can be retrieved at any time in a structure. | Only one member can be accessed at a time in a union. |

**PART-B**

**Answer any three Questions 3 x 12 = 36**

1. What is structure? Explain in detail about structure with suitable example.

Structure is a collection of variables (can be of different types) under a single name.

* [How](https://www.programiz.com/c-programming/c-structures#define) to Define Structure
* [Create structure variable](https://www.programiz.com/c-programming/c-structures#create)
* [Access members of a structure](https://www.programiz.com/c-programming/c-structures#access-members)
* [Example: Add two distances](https://www.programiz.com/c-programming/c-structures#example)
* [Keyword typedef](https://www.programiz.com/c-programming/c-structures#typedef)
* [Nested structures](https://www.programiz.com/c-programming/c-structures#nested)

1. Explain in detail about pointer, declaration, initialization and its accessing with an example.

**Pointer** is a programming language object that stores the memory address of another value located in computer memory. A **pointer**references a location in memory, and obtaining the value stored at that location is known as dereferencing the **pointer**.

**Declaration** is a language construct that specifies properties of an identifier: it declares what a word (identifier) "means".

**Initialization** is the assignment of an initial value for a data object or variable. The manner in which **initialization** is performed depends on **programming** language, as well as type, storage class, etc., of an object to be**initialized**.

 In **Access**, **programming** is the process of adding functionality to your database by using **Access** macros or Visual Basic for Applications (VBA) code. For example, suppose that you have created a form and a report, and you want to add a command button to the form that, when clicked, opens the report.

1. What are all the storage classes available in ‘C’ and explain those in details with suitable examples?

**Storage Classes**

Every C variable has a storage class and a scope. The storage class determines the part of memory where storage is allocated for an object and how long the storage allocation continues to exist. It also determines the scope which specifies the part of the program over which a variable name is visible, i.e. the variable is accessible by name. They are four storage classes in C are automatic, register, external, and static.

* [Automatic Variables](http://www-ee.eng.hawaii.edu/~tep/EE160/Book/chap14/subsection2.1.1.1.html" \l "SECTION0011100000000000000)

* [Register Variables](http://www-ee.eng.hawaii.edu/~tep/EE160/Book/chap14/subsection2.1.1.2.html" \l "SECTION0011200000000000000)

* [External Variables](http://www-ee.eng.hawaii.edu/~tep/EE160/Book/chap14/subsection2.1.1.3.html" \l "SECTION0011300000000000000)

* [Variable Definition vs Declaration](http://www-ee.eng.hawaii.edu/~tep/EE160/Book/chap14/subsection2.1.1.4.html" \l "SECTION0011400000000000000)

* [An Example: Lexical Scanner](http://www-ee.eng.hawaii.edu/~tep/EE160/Book/chap14/subsection2.1.1.5.html" \l "SECTION0011500000000000000)

* [Static Variables](http://www-ee.eng.hawaii.edu/~tep/EE160/Book/chap14/subsection2.1.1.6.html" \l "SECTION0011600000000000000)

* [Storage Class for Functions](http://www-ee.eng.hawaii.edu/~tep/EE160/Book/chap14/subsection2.1.1.7.html" \l "SECTION0011700000000000000)

* [Stack vs Heap Allocation](http://www-ee.eng.hawaii.edu/~tep/EE160/Book/chap14/subsection2.1.1.8.html" \l "SECTION0011800000000000000)

1. What are all the preprocessor available in ‘C’ and explain those in details with suitable examples?

* Include Header Files
* Macros
* Conditional Compilation
* Diagnostics
* Line Control
* Progmas
* Preprocessor Output