Assignment 5: Structures & Pointers

[Q.1]: Differentiate between pass by value and pass by reference with an example.

Ans:

Pass by Value	Pass by reference.
Parameter value is passed to invok function.	Parameter address is passed to invok function.
Any change in value made inside function has no effect in passed value.	Any change in value inside function has change in passed value.
passes copy of variable.	passes the variable itself.
Directly value is passed.	Pointer has to be defined.

Eg: int main()
{ int n=10;
print(n);
}
print(int n)
{ ....; }
	Eg: void main()
	{ int n=10; int *p;
	*p = &n.
	print(&n);
	}
	print(int * x)
	{ ....; }

[Q.2]: What is pointer variable? What are the advantages of using pointer?

Ans:

Pointers are ~~de~~ the special variables containing address of any other variable. It is derived datatype. The advantages of using pointers:

i): Allows dynamically memory at runtime.

ii): Allows accessing data & modification is easier.

iii): Enables multiple functions to access data.

<Q.3>: What is the difference between array and pointer variable? In what way are they similar?

Ans:

Array	Pointer Variable:
Array allocates a contiguous block of memory to store fixed number of elements.	Pointer variable doesn't have memory allocation as array.
Size of array determined at compile time and remains constant throughout lifetime.	Pointer size is the size of the datatype and is known & fixed.
An array cannot be reassigned to ~~anott~~ point to another location.	A pointer can be reassigned to point to a different address.

Similarities:

i) Both array and pointer variable are used to access and manipulated data indirectly.

ii) Array and pointer are closely related. The array name is the pointer for its base address.

iii) In passing to function, both undergo pass by reference.

<Q.4>: What is structure? How is it different from union?

Ans:

Structures are users defined datatypes which is the collection of one or more datatypes grouped together under a single name.

Structure	Union:
Each memory has its own memory space.	All members share same memory space.
Total memory is the size of sum of all members.	Size of memory is determined by largest member.
Each member accessed individually and change in one member doesn't affect another.	Only one member is accessed at a time and change in one member affects another.
Used for representing complex data structure.	Used for scenarios where memory efficiency crucial.

<Q.5>: What are DMA? Why are they important? How do we apply?

Ans:

DMA is the process of allocating memory at run time.

They are important because:

i) Efficient memory usage.

ii) User can allocate and deallocate memory as pleased.

DMA is done by using DMA functions.

a) malloc: It allocates a block of memory from heap section. Eg: (int*) malloc (100 * sizeof(int));

b) calloc: It allocates multiple blocks of memory in contiguous form.
Eg: (int*) calloc (5, sizeof(int));

c) realloc: It alters the size of previously allocated block.
Eg: realloc (p, 7 * sizeof(int));

d) free: It deallocates the memory allocated using DMA
Eg: free (p);