Texity Systems
Test taken in
Aug. 2011

.What is static memory allocation and dynamic memory allocation?

Static memory allocation: The compiler allocates the required memory space for a declared variable.By using the address of operator,the reserved address is obtained and this address may be assigned to a pointer variable.Since most of the declared variable have static memory,this way of assigning pointer value to a pointer variable is known as static memory allocation. memory is assigned during compilation time.

Dynamic memory allocation: It uses functions such as malloc( ) or calloc( ) to get memory dynamically.If these functions are used to get memory dynamically and the values returned by these functions are assingned to pointer variables, such assignments are known as dynamic memory allocation.memory is assined during run time.

What is a pointer variable?

A pointer variable is a variable that may contain the address of another variable or any valid address in the memory.

What is a pointer value and address?

A pointer value is a data object that refers to a memory location. Each memory location is numbered in the memory. The number attached to a memory location is called the address of the location.

What is a null pointer?

There are times when it's necessary to have a pointer that doesn't point to anything. The macro NULL, defined in , has a value that's guaranteed to be different from any valid pointer. NULL is a literal zero, possibly cast to void* or char*. Some people, notably C++ programmers, prefer to use 0 rather than NULL.

The null pointer is used in three ways:

1) To stop indirection in a recursive data structure
2) As an error value
3) As a sentinel value

What is a void pointer?

A void pointer is a C convention for a raw address. The compiler has noidea what type of object a void Pointer really points to. If you write
int *ip;
ip points to an int. If you write
void *p;
p doesn't point to a void!
In C and C++, any time you need a void pointer, you can use another pointer type. For example, if you have a char*, you can pass it to a function that expects a void*. You don't even need to cast it. In C (but

not in C++), you can use a void* any time you need any kind of pointer, without casting. (In C++, you need to cast it).
A void pointer is used for working with raw memory or for passing a pointer to an unspecified type.
Some C code operates on raw memory. When C was first invented, character pointers (char *) were used for that. Then people started getting confused about when a character pointer was a string, when it was a character array, and when it was raw memory.

When should a far pointer be used?
Sometimes you can get away with using a small memory model in most of a given program. There might be just a few things that don't fit in your small data and code segments. When that happens, you can use explicit far pointers and function declarations to get at the rest of memory. A far function can be outside the 64KB segment most functions are shoehorned into for a small-code model. (Often, libraries are declared explicitly far, so they'll work no matter what code model the program uses.) A far pointer can refer to information outside the 64KB data segment. Typically, such pointers are used with farmalloc() and such, to manage a heap separate from where all the rest of the data lives. If you use a small-data, large-code model, you should explicitly make your function pointers far.

How are pointer variables initialized?
Pointer variable are initialized by one of the following two ways
- Static memory allocation
- Dynamic memory allocation

Difference between arrays and pointers?
- Pointers are used to manipulate data using the address. Pointers use * operator to access the data pointed to by them
- Arrays use subscripted variables to access and manipulate data.

Array variables can be equivalently written using pointer expression.

What is the difference between declaring a variable and defining a variable?
Declaring a variable means describing its type to the compiler but not allocating any space for it. Defining a variable means declaring it and also allocating space to hold the variable. You can also initialize a variable at the time it is defined.

Differentiate between an internal static and external static variable?
An internal static variable is declared inside a block with static storage class whereas an external static variable is declared outside all the blocks in a file.An internal static variable has persistent storage,block scope and no linkage.An external static variable has permanent

storage,file scope and internal linkage.

What is the difference between a string and an array?

An array is an array of anything. A string is a specific kind of an array with a well-known convention to determine its length.

There are two kinds of programming languages: those in which a string is just an array of characters, and those in which it’s a special type. In C, a string is just an array of characters (type char), with one wrinkle: a C string always ends with a NUL character.

The “value” of an array is the same as the address of (or a pointer to) the first element; so, frequently, a C string and a pointer to char are used to mean the same thing.

An array can be any length. If it’s passed to a function, there’s no way the function can tell how long the array is supposed to be, unless some convention is used. The convention for strings is NUL termination; the last character is an ASCII NUL (”) character.

What is the difference between a string and an array?

An array is an array of anything. A string is a specific kind of an array with a well-known convention to determine its length.

There are two kinds of programming languages: those in which a string is just an array of characters, and those in which it’s a special type. In C, a string is just an array of characters (type char), with one wrinkle: a C string always ends with a NUL character.

The “value” of an array is the same as the address of (or a pointer to) the first element; so, frequently, a C string and a pointer to char are used to mean the same thing.

An array can be any length. If it’s passed to a function, there’s no way the function can tell how long the array is supposed to be, unless some convention is used. The convention for strings is NUL termination; the last character is an ASCII NUL (”) character.

What is an argument? Differentiate between formal arguments and actual arguments?

An argument is an entity used to pass the data from calling function to the called function. Formal arguments are the arguments available in the function definition. They are preceded by their own data types. Actual arguments are available in the function call.

Texity Systems

1] Difference between malloc & new

2] Diff- between Interface & Abstract class

3] Different features provided by OOP

4] Structure of JRE

5] Inheritance in Java [multiple]

6] Project work with role played in team

7] Abstract class, definition & its use.

8] Can abstract class have constructor

9] Difference overriding & overloading

10] Try-catch blocks in exception handling

11] what is mean by object persistence

12] ~~w us~~ If we have two interfaces with
same method signature,
can we write/define both method
in a class which implements the
two interfaces

13] Resume Dependent [skill areas]

*] In - financial Technologies

↳ concept of pointer

↳ Array

↳ Link list

↳ Implementation of Doubly
linked list — Insert
— Delete

[complete - code]

↳ malloc — use

↳ memory leak

↳

Find the digits after decimal point.

```
a=45.0206
b=long(a);

x = .02060 x10
    0.206
0
    0.206 x10
    2.060
2   .06 x10
    .6000
0   .6 x10
    6
6   [0]000
```

```
float a,c,x;
long b=(long)a;
c= a-b.
do { x=c*10;
     c= x-(long)x;
} while (c>0)
```

```
sprintf(ch[i] "%f", a[i]);
```

* Convert string entered to long

```
# include <stdlib.h>
# include <stdio.h>.
int  main(void)
{
  char  num1[80]; ~~num2[80]~~
  printf ("Enter first:");
  gets (num1);
   prinf ("The num is : %ld", atol(num1));
  return 0;

}
```

```
int main()
{char sz[80]="2601"; char *p; long li;
 ~~li = strtol(str)~~;
 printf ("Enter an long value: ");
  ~~gets(str)~~
  li=strtol (str,&p,0);
  printf("%ld", li );
  return 0;
}
```