**QUIZ 1**

* If you like to see accurate debugging information, which of the following program processing would you recommend? **Interpretation**
* What is the main reason of applying two-step translation of high level programming language? **One compiler for all machines**
* If your application is composed of multiple modules (programs), which of the following program processing would you recommend? **Compilation**
* If your program was designed to print “Hello World” ten (10) times, but during execution, it printed eleven (11) times. What type of error is it? **Semantics Error**
* Which command will have a loop when expressed in a syntax graph?  **switch**
* Which commands (constructs) do NOT have a loop when expressed in syntax graphs? Select all that apply. **while, if/else, for**
* What does the | (pipe) symbol in a BNF rule mean**? It is an or statement. Choose one of the options.**
* How many different identifiers can the following BNF ruleset generate? **more than 26 (infinite)**
* Given this snippet of code in C,
  + char alpha = 'a';
  + int numeric = alpha + 10;
  + which of the following statement is correct: **Syntactically correct, but contextually incorrect**
* If a program contains an error that divides a number by zero at the execution time. **This error is a semantic error**
* Which of the following cannot be checked by an imperative or object-oriented compiler. (Check all that applies) **Semantics**
* Which commands (constructs) do NOT have a loop when expressed in syntax graphs? Select all that apply  **for ( <init-expr>; <test-expr>; <increment-expr> ) {<statements>} , while (condition) do {statements;}, if-then-else**
* Can the identifier "base\_variable" be created from the following BNF ruleset?
  + <char> ::= a | b | c | ... | s | ... | x | y | z
  + <identifier> ::= <char> | <char> <identifier>
  + **No - there is an underscore in the identifier name that cannot be generated.**
* Java uses which of the following program processing? **Two Step Translation with Intermediate Code -- (Compilation + Interpretation)**
* How many different identifiers can the following BNF ruleset generate?
  + <char> ::= a | b | c | ... | x | y | z
  + <identifier> ::= <char> | <char> <identifier>
  + **More than 26**
* During compilation, linker is used for **resolving external references (bring in code from other libraries).**
* For the following BNF ruleset, which are terminal symbols? (Check all that apply.)

<char> ::= a | b | c | ... | x | y | z

<identifier> ::= <char> | <char> <identifier>

**Y, A**

* A programming languages can be broken down into four structural layers. Identify these layers. (Check all that applies.)
  + **Syntactic, Contextual, Semantics**

**QUIZ 2**

* Implicit type conversion is commonly referred to as **Coercion.**
* Explicit type conversion is commonly referred to as **Casting.**
* Type checking happens during compilation phase, it reinforces which of the following structural layer? **Contextual**
* Where is the main() function located in a C or C++ program? **outside any class**
* Sort orthogonality 1 and sort orthogonality 2 implies compositional orthogonality. **True**
* In the C-Style input function scanf(“%d”, &i); What does the character “&” mean? **scanf takes the address of an variable as its parameter.**
* In C++, what function can be used to input a string with spaces? **cin.getline(...);**
* Given the following code, what is the expected value for z? **9**
  + #define func(x, y) (x > y) ? y++ : x++
  + int main()
  + {
    - int x = 10;
    - int y = 9;
    - int z = func(x, y);
  + }
* (True or False) Sort orthogonality 1 and sort orthogonality 2 implies compositional orthogonality. **True**
* Which of the following statement is correct if a language is strongly typed. (Check all that applies.) **Each name in a program has single type associated with it, Type errors are always reported**
* Which implementation of a function has potentially the best performance in terms of execution speed? **Macro**
* Which of the following orthogonality describe this example: If a block allows one statement, it should allow zero or more statements within that same block**. Number Orthogonality**
* Macros-Processing takes place during which phase? **Pre-processing**
* Given the following code, what is the expected value for z?**10**
  + #include <stdio.h>
  + #define func(x, y) (x > y) ? y : x
  + int main()
  + {int x = 10;
  + int y = 9;
  + int z = func(++x, y++);}
* Given the following code, what is the expected value for z?**11**
  + #include <stdio.h>
  + #define func(x, y) (x > y) ? y : x
  + int main()
  + {int x = 10;
  + int y = 9;
  + int z = func(x++, ++y);}
* Assume a function requires 20 lines of machine code and will be called 10 times in the main program. You can choose to implement it using a function definition or a macro definition. Compared with the function definition, macro definition will lead the compiler to generate, for the entire program, **a longer machine code but with shorter execution time**

**QUIZ 3**

* Given this snippet of code, what is the value of x after executing the last statement? **10**
  + int x = 10, \*y;
  + y = &x;
  + y = y + 1;
  + \*y = 100;
* Given this snippet of code, what is the value of x after executing the last statement? **100**
  + int x = 10, \*y;
  + y = &x;
  + \*y = 100;
  + y = y + 1;
* Given this snippet of code, what is the value of z after executing the last statement?
  + int x = 10, \*y, \*\*z;
  + z = &y;
  + y = &x;
  + \*y = 100;
  + Answer: **None of the above(10, 100, 1000)**
* include <stdio.h>
  + main() {
  + int m = 30, n = 40, \*p, \*q, \*\*r;
  + p = &m;
  + \*p = 60;
  + q = &n;
  + \*q = 80;
  + r = &p;
  + \*\*r = 100;
  + printf("%d\n", m); // 1st printf statement
  + printf("%d\n", n); // 2nd printf statement
  + m = 30;
  + n = 90;
  + printf("%d\n", \*\*r); // 3rd printf statement
  + }
  + The output of the 1st printf statement is **100**.
  + The output of the 2nd printf statement is **80**.
  + The output of the 3rd printf statement is **30**.
* C/C++ has 2 pointer operators, which operator will return the address of variable? **Ampersand(&)**
* Which of the following types is a C++ type, but NOT a C type?**bool**
* C/C++ has 2 pointer operators, which operator represents the name of the address? **Asterisk(\*)**
* What type of value are stored within a pointer type? **a integer value, which represents the address.**
* Given a C declaration: char a[] = "Hello World"; What can be used as the initial address of array a[]? Select all correct answers. **&a[0], a**
* Which of the following C declarations (contextual level) will cause a compilation error**? int d[];**
* Which of the following C assignment statements (assign a value to a variable at the semantic level) will NOT cause a compilation error? Assume the array has been declared as: char a[5]; **a[0] = 'h';**
* Assume a variable is declared in a block of code within a pair of curly braces. **The scope of the variable starts from its declaration point and extends to the end of the block.**
* Multiple pointers can reference the same objects. **True**
* A data type defines the **values and operations allowed**
* During data declaration, a name is binded to a memory location, what else can be identify as part of this process? **Data Type,Scope, Qualifier**
* Given the C declaration: char s1[4], s2[ ] = "hello"; if a string copy function strcpy(s1, s2) is executed, what will happen? **The result is s1 will contain the string "hell", and the following two byte locations will contain 'o' an '\0'.**

**QUIZ 4**

* Given the following code
  + char \*p = "hello", \*s = "this is a string";
  + p = s;
  + printf("%s\n", p);
  + What will happen? **It prints: this is a string**

char \*p = "hello", \*s = "this is a string";

strcpy(p, s);

printf("%s\n", p);

**A run time error may occur at this line: strcpy(p, s);**

* Given the following code
  + char a[2][3] = { { 'c', 'a', 't'}, { 'd', 'o', 'g'} };
  + int i, j;
  + for (i = 0; i<2 ; i++) {
  + for (j = 0; j<3; j++)
  + printf("%c", a[i][j]);
  + }
  + What will happen? It prints: **catdog**
* Given the following code
  + char a[] = {'c', 'a', 't', '\0'};
  + char \*p = a;
  + while (\*p != 0) {
  + \*p = \*p + 1;
  + printf("%c", \*p++); // \*p++ is the same as \*(p++)
  + }
  + What will happen? It prints: **dbu**
* Given the following code
* char a[2][4] = { { 'c', 'a', 'r', 'b' }, { 'i', 'k', 'e', '\0' } }; char \*p = &a[0][0]; while (\*p != '\0') { printf("%c", \*p); p++; }
  + What will happen? It prints: **carbike**
* Given the declaration: char A[3] = {'C', 'a', 'r'}, \*p = A; what statement prints character a? **printf(“%c”,\*(++p));**
* Consider the following snippet of code in a 32-bit computer.
  + #define max 10
  + struct contact {
  + char name[30];
  + char email[30];
  + int phone;
  + };
  + struct contact A[max];
  + int tail = 0;
  + What is the size of variable x in bytes? **64**
  + Which statement can read a name into the name field of the structure? **scanf("%s", contactbook[tail].name);**
* What is the main problem of using an array of structures to store data records such as contact list? **The total length of the list must be determined in advance.**
* The tail variable in an array of structures is used for indicating (Select all that apply)
  + **how far the search() function should go in searching the items of the array.**
  + **where to insert a new item, if the items in the array do not need to be sorted**.
* What C/C++ constant variable can potentially be modified during the execution? **const x = 5;**
* What data types in C have the same data range, assuming that the computer used is a 32-bit computer? **Pointer type and unsigned int**
* Given the following snippet of code, answer the following two questions based on the code:
  + typedef enum {Sun, Mon, Tue, Wed, Thu, Fri, Sat} days;
  + days x = Mon, y = Sat;
  + while (x != y) { x++; }
  + Y++;
  + printf("x = %d, y = %d", x, y);
  + 1. What value will be printed for variable x? [x].
  + 2. What value will be printed for variable y? [y] **6, 7**
* Given the following snippet of code, answer the following two questions based on the code:
  + typedef enum {red, amber, green} traffic\_light;
  + traffic\_light x = red, y = green;
  + while (x != y) { x++; }
  + y++;
  + printf("x = %d, y = %d", x, y);
* What value will be printed for variable x? Please choose: **2**.
* What value will be printed for variable y? Please choose: **3**
* The tail variable in an array of structures is used for indicating
  + **Where to insert a new item, if the items in the array do not need to be sorted**
  + **How far the search() function should go in searching the items of the array**
* Consider the following snippet of code in a 32-bit computer.
  + #define max 10
  + struct contact {
  + char name[30];
  + char email[30];
  + int phone; };
  + struct contact A[max];
  + int tail = 0;
  + Which statement can read a phone number into the phone field of the structure?
  + **scanf("%d", &contactbook[tail].phone);**

**QUIZ 5**

* How do you properly delete the first node of a linked list pointed to by head, assuming that the linked list is not empty and temp is another pointer of the same type?
  + **temp = head;**
  + **head = head->next;**
  + **free(temp);**
* Given the information below, which of the following snippet of codes will insert a new node in the second place in the linked-list. Assume the linked-list contains already at least one node.
* struct Terminal {
  + char name[30];
  + char location[32];
  + struct Terminal\* next;
* } \*head, \*p, \*q;
  + **p = (struct contact \*) malloc(sizeof(struct contact));**
  + **...**
  + **p->next = head->next;**
  + **head->next = p;**
* Assume that the search function of a linked list is specified by
  + struct Terminal\* search();
  + What values can the search function return? Select all correct answers.
  + **0, the address of a terminal node**
* When will the buffer be created? **When the file operation fopen is performed.**
* The reason that we use a buffer between the disk and the memory is **Disk access speed is low, and disk is read and written large block of data.**
* The reason that we need to call fflush() or cin.ignore() is because the previous **input leaves a character in the file buffer.**
* Given the information below, how will you access the name for a terminal node pointed to by x? **x->name**;
* struct Terminal {
  + char name[30];
  + char location[32];
  + struct Terminal\* next;
* } \*x;
* Assume this is a 32-bit environment, what is the size of x? **4 bytes**
* struct Terminal {
  + char name[30];
  + char location[32];
  + struct Terminal\* next;
* } \*x;
* Assume pointer variable p points to node x, and node x's next pointer points to node y. What does free(p) operation mean? **return the memory held by x to the free memory pool.**
* When is padding required for a structure type variable? **When the structure contains a word-type variable, such as integer, float, and pointer, and the total number of bytes is not a multiple of four.**
* Which of the following coding practice can lead to a higher memory efficiency (use fewer padding bytes) in defining a structure? **Keep the character type variables together.**
* The reason that we need to call fflush() or cin.ignore() is because the previous **input leaves a character in the file buffer.**
* The size(number of bytes) of a structure-type variable can be changed by the following factors.
  + **Changing the computer from a 32-bit to a 64-bit processor, adding a member into the structure, changing the orders of the members in the structure**
* Consider the following snippet of code in a 32-bit computer.
  + struct contact {
  + char name[30];
  + int phone;
  + char email[30];
  + } x;
  + What is the size of variable x in bytes?
  + **68**
* What parameters are required when performing file operations fread and fwrite? **Destination, source, number of items, item size**

**QUIZ 6**

* Consider an array, a linked list, and a binary search tree. Which data structure requires fewest comparisons in average to search an element stored in the data structure? **binary search tree**
* The search algorithm will be more efficient if a binary search tree is a **balanced binary tree.**
* Which recursive functions require to define more than one size-m problems? **Hanoi tower, Greatest Common Divisor gcd function**
* In the hanoi towers function, what part of the code represent step 4: Construction of size-n problem from size-(n-1) problems?
  + **hanoitowers(n-1, S, D, M);**
  + **hanoitowers(1, S, M, D);**
  + **hanoitowers(n-1, M, S, D);**
* What is the time complexity of insertion sort algorithm? **O(n\*n)**
* Given the snippet of codes, identify the passing mechanism used for y (in func). **call-by-alias**
* void func(int \*x, int &y)
* `{
  + \*x = \*x + y;
  + y = 2;
* }
* Which parameter passing mechanism does not have side-effects? **call-by-value**
* Given the snippet of codes, what is the expected output?
* void func(int \*x, int y)
* {
  + \*x = \*x + y;
  + x = 10;
* }
* void main()
* {
  + int x = 10, y = 10;
  + func(&x, y);
  + printf("x: %d, y: %d", x, y);
* } **x: 20, y: 10**
* When inserting a data into a binary search tree, the data should be **inserted at the position to keep the entire tree as a binary search tree.**
* If you want to change the insertion sort function with a size-(n-1) problem, as discussed in the lecture, to a merge sort function, where do you need to make changes? **In the definition of size- m problem, In the code that constructs the solution of size-n problem from the size-m problem.**
* The function searching a binary search tree can be easily implemented using **a recursive function with two recursive calls.**
* The data stored in a binary search tree is sorted, if the tree is traversed in **inorder.**
* The complexity of searching an arbitrary binary search tree is the order of **O(n)**

**Notes:**

What are the four-steps abstract approach for solving recursive problems?

1.Formulate the size-n problem.

2.Find the stopping condition and the corresponding return value.

3.Formulate the size-m problem and find m. In many cases, m = n - 1;

4.Construct the solution of size-n problem from size-m problem.

