CSE12 Quiz2

1) What is a void pointer?
   Pointer without type. Pointer to anything
2) What can't you do with a void pointer?
   dereference

3) A function is implemented via what stack operation to the run-time stack?
   Push (can someone explain 3 and 4) When you call a function, you push in a new stack frame
4) Returning from a function is implemented via what stack operation to the run-time stack?
   Pop (Explanation on Piazza, Here!)

5) Besides not knowing how much memory you need, what the other main situation discussed in class when you must dynamically allocate memory in C and C++?

   - **When you need a lot of memory. Stack is relatively small, so you program might crash if you put too much memory onto stack**
   - **When the memory must remain accessible after the function returns. Stack memory gets destroyed when function and returns. Dynamic memory is available until you call free()**
   - ***We want the memory to exist beyond method execution***

6) In C and C++, what is the number value of false?
   0
7) What In C and C++, what is the number value of true?
   Non zero
8) In C and C++, what is the number value of NULL?
   zero

9) In what section of memory do parameters exist?
   RTS
   In what section of memory do local variables exist?
   RTS

10) In what section of memory do static variables exist?
    data

11) What is the return type of "malloc"?
    void pointer
12) What is the type of the parameter to "free"?
    void pointer

13) How does the parameter to "free" relate to the return value of "malloc"?
    the same
14) What is the unit of allocation for the parameter sent to "malloc"?
    bytes

15) What is the method call to give back memory allocated in C?
free

16) A "stack" data structure has the property: First In ________(first in first out is Queue)__
last out

17) A "stack" data structure has the property: Last In __________
first out

18) What keyword is used to dynamically allocate memory in Java and C++?
new

19) What keyword is used to dynamically deallocate memory in C++?
delete

20) stderr is used to display what two kinds of messages:
error and debug

21) Write a function with the purpose of assigning a long that exists elsewhere to 0.

```
Void func(long * lp) {
      *lp = 0;                    /*SHOULDN'T IT BE lp = 0?*/ /that would be assigning the address
to 0*/ /* I believe "lp = 0" would be assigning a copy of the variable to 0, not the actual long
*/

}
```

22) Write a function with the purpose of assigning a long pointer that exists elsewhere to 0.

```
Void func(long ** lpp ) {
      *lpp = 0;
}
```

23) Write a main function that has a function call to assign a local long variable to 0.

```
Main () {
            Long lll;
            Func(&lll);
}
```

24) Write a main function that has a function call to assign a local long pointer variable to 0.

```
Main() {
   Long * lp;
   Func(&lp);
}
```

25) Are the following two types the same or different?
pointer to pointer to long
pointer to long pointer
same

26) Showing variable names, known memory addresses and including arrows where appropriate, draw the RTS showing a main method with a local long variable, lll, calling a method that assigns lll to 0. The name of the parameter is lp. The address of lll is 1000.

lp:	1000
	...
1000:lll:	0

RTS

Someone please fix it if it's wrong (should have an arrow there?)

There should be an arrow from 1000 to the 0 which originally was a random value.
Gary said on piazza that "there will be no drawing of hw3 stacks or hw5 lists". So I think this question may not be on the exam.evidence here
But this question is neither hw3 stacks nor hw5 lists...

27) Showing variable names, known memory addresses and including arrows where appropriate, draw the RTS showing a main method with a local long pointer variable, lp, calling a method that assigns lp to 0. The name of the parameter is lpp. The address of lp is 1000.

lpp:	1000
	...
1000:lp:	0

RTS

28) Showing variable names, known memory addresses and including arrows where appropriate, draw the RTS for: void func (long * lp) { *lp = 0; } main() { long lll; func (&lll);} The address of lll in main is 1000. (the same as 27)

29) What is the type of 0?
ALL
30) Is 0 of type void pointer?
YES
31) Is 0 of type int pointer?
YES

32) In the expression: xzy = sizeof (Stack), is sizeof (Stack) a typical function call?"
No, because Stack is a type; not param

33) Excluding validity checks and return values, write the body code for the push function of hw3.

```
Long push (Stack* this_Stack, long  item) {

  Long sp = this_Stack[SPI];
  this_Stack[sp] = item;
  this_Stack[SPI] ++;
}
```

34) Excluding validity checks and return values, write the body code for the pop function of hw3.

```
Long pop (Stack* this_Stack, long  * item) {

        long sp = this_Stack[SPI];
        *item = this_Stack[sp-1];
        this_Stack[SPI]--;
        return sp;
}
```

35) Excluding validity checks and return values, write the body code for the top function of hw3.

```
Long top(Stack* this_Stack, long* item) {

        long sp = this_Stack[SPI];
        *item = this_Stack[sp - 1];
        return sp;
}
```

36) Excluding validity checks and return values, write the body code for the empty_Stack function of hw3.

```
Long empty_Stack (Stack* this_Stack) {
        this_Stack[SPI] = 0;
}
```

37) What condition is checked to determine if the stack is empty in hw3?
If(this_Stack[SPI] == 0)

38) What condition is checked to determine if the stack is full in hw3?
If (this_Stack[SPI] == this_Stack[SSI])

39) What are the two responsibilities of the delete_Stack function of hw3?
Deallocate the stack; set the pointer to NULL

40) What are the two responsibilities of the new_Stack function of hw3?
Allocate memory; initialize the memory(SPI, SSI , SCI)

41) What is the Object-Oriented term describing each of a collection of functions defined within the same file in C each with a common first parameter?
Member function

42) What is the Object-Oriented term describing the new_Stack method of hw3?
constructor

43) What is the Object-Oriented term describing the delete_Stack method of hw3?
destructor

mainStack.push(10); //java code

44) Which stack methods of hw3 need to check if the stack is full before the requested operation can be completed successfully?
Push, <u>isfull_Stack</u>
45) Which stack methods of hw3 need to check if the stack is empty before the requested operation can be completed successfully?
Pop, top, isempty_Stack

46) In the push function, what is the name and type of the first parameter?
Type: Stack*; name : this_Stack
47) In the pop function, what is the name and type of the first parameter?
Type: Stack*; name: this_Stack
48) In the top function, what is the name and type of the first parameter?
Type: Stack*; name: this_Stack

49) When calling free, is the caller giving up access to or authority to access memory or both?
Giving up authority
50) When assigning a pointer to null, is the caller giving up access to or authority to access memory or both?
Giving up access
51) In C, what are the three illegal kinds of declarations?
Can't return an array
Can't return a function
<u>an array of function</u>
52) With parallel arrays, two or more arrays share the same ________
Index
53) In C, in what kind of file do you list public information? **.h header file**
54) In C, in what kind of file do you list private information? **.c file**
55) In C, in what kind of file do you list hidden information? **.c file**

56) Generally speaking, what goes in a .h file? (an answer of declarations and type definitions is too specific)
public information

57) Name the three binary masking bit manipulation operations **AND OR XOR (& | ^)**
58) Intuitively speaking, ANDing with ones gives ____**same (original number)**________
59) Intuitively speaking, ANDing with zeros gives _____**zero**_______
60) Intuitively speaking, ORing with ones gives ________**1s**____
61) Intuitively speaking, ORing with zeros gives __________**same**__
62) Intuitively speaking, XORing with ones gives _______**flip**_____
63) Intuitively speaking, XORing with zeros gives _________**same**___

64) What is the English word to describe XORing with one continuously? **Toggle**
65) XORing a value with itself gives ______**0**

66) In 8 bits, what is the result of ~01010101? **10101010**
67) In 8 bits, what is the result of 01010101 & 00001111? Source & mask **00000101**

68) In 8 bits, what is the result of 01010101 | 00001111? **01011111**
69) In 8 bits, what is the result of 01010101 ^ 00001111? **01011010**
70) How do you describe the result of the following three operations?
xxx ^= yyy; yyy^=xxx; xxx^=yyy;

xxx = 0000 yyy = 1111
0000^1111 => 1111 stores in xxx
yyy ^= xxx;
1111^1111 => 0000 stores in yyy
xxx^=yyy
1111^0000 => 1111 stores in xxx

X Y SWAPs

71) T or F: ANDing is used to extract bits. **T**
72) T or F: ANDing is used to set bits. F
73) T or F: ANDing is used to clear bits. T
74) T or F: ANDing is used to flip bits. F

75) T or F: ORing is used to extract bits. F
76) T or F: ORing is used to set bits. T
77) T or F: ORing is used to clear bits. F
78) T or F: ORing is used to flip bits. F

79) T or F: XORing is used to extract bits. F
80) T or F: XORing is used to set bits. F
81) ;rT or F: XORing is used to flip bits. T

82) Shifting bit to the right has the effect of ____________**divide by power of two**
83) Shifting bit to the left has the effect of ____________**multiply by power of two**

84) What is the operator used to shift bits to the left? **<<**
85) What is the operator used to shift bits to the right? **>>**

86) In your calculator of hw4, what is the best word to describe the character that intopost will "unget"?
digit
87) In your calculator of hw4, what is the best word to describe the character that decin will "unget"?
non-digit
88) In your intopost code of hw4, the values pushed to a stack originate from what two functions?
decin, setupword

Using English words, describe the following declarations:
(http://ieng9.ucsd.edu/~cs30x/rt_lt.rule.html)
89) int (*x)[][]; **pointer to an array of int arrays**
90) int (*x)()[]; **pointer to function returning array of ints(ILLEGAL)**
91) int (*x[])(); **an array of pointers pointing to functions returning int**

92) int []x[](); **an array of arrays of functions returning an int (ILLEGAL)**
93) int (*x())[]; **a function returning a pointer to an array of ints**
94) int x()[](); **a function returning an array of functions returning int (ILLEGAL)**
95) int *(*x)[]; **a pointer to an array of int pointers**
96) int (*x())(); **a function returning a pointer to a function returning int**
97) int (*x[])[]; **an array of pointers pointing to array of ints**
98) long (*x[])(long,long); **array of pointer to functions taking two longs returning long**

99) Assuming malloc returned 2000, draw a hw3 stack created with a size of 10 after pushing 25, 50, 75, 100 and popping twice.
2000: [an array of 10 items] [1] [10] [2] | [25] [50] [75] [100]

SCI SSI SPI

-3	-2	-1	0	1	2	3	4	5	6	7	8	9
1	10	2	25	50	75	100						

Shouldn't the stack be 2000: [an array of 10 items] [1] [10] [2] | [25] [50] since it was popped twice? No, since you don't change the values at the popped indices, only change the SPI aka the next *available* index, so the next available index would be 2 but the old values still are there in the memory, and you know them

zJ

100) In Java or C++, a static method is one called without _________
object

101) If there is 0 instance of a class, how many instances are there of one of its static data fields?
one

102) If there are 1000 instances of a class, how many instances are there of one of its static data fields?
one

103) Is a static data field part of the "sizeof" an object? **no**
104) Is a static data field allocated in adjacent memory to objects of that class? **no**
105) In C, a static method not inside a class is one that can be called by what other methods?

→ **any methods in the same file below the declaration of that method**
// why are these methods below the declaration of that methods?
I GUESS, BEFORE THE DECLARATION OF THE STATIC METHOD, YOU DO NOT EVEN HAVE ONE….HOW CAN YOU CALL IT???

106) What is the value of xyz at the end of the fifth execution of the following function?
```
void func() {static int xyz = 10; xyz++;}
```
15

107) What is the value of xyz at the end of the fifth execution of the following function?
```
void func() {int xyz = 10; xyz++;}
```
11

108) In C and C++, unless otherwise initialized, what is the initial value of all static variables?

0

109) Can static methods be accessed by name outside the file in which they are declared?

no