

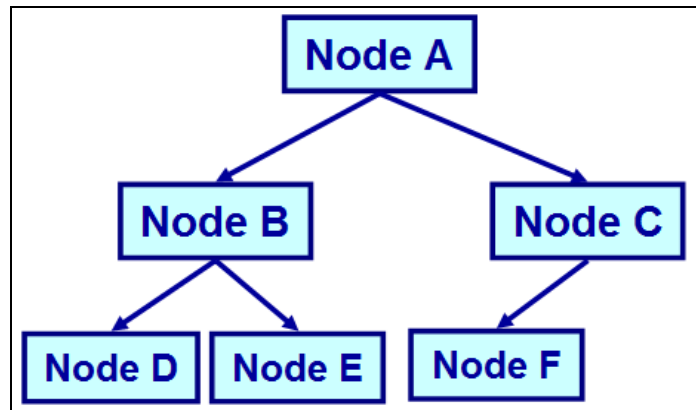
Quiz #3**Name:** _____**Student Number:** _____**Signature:** _____**Instructions**

1. Fill in your Name, Student Number, and signature above.
2. This is a closed book Quiz. No electronic or paper aids permitted.
3. Do not open this test booklet until instructed to do so.
4. Clearly indicate if some part of your work is not to be marked. Add as many comments as needed to provide a clear response.
5. You may answer the questions in any order you want.
6. Raise your hand if you have a question. The instructor will come over to assist you.
7. Copying from or communicating with a neighbor or with anyone directly or electronically will result in both students receiving a zero and may result in further disciplinary action by the school and or university administration.
8. The total number of points for this Quiz is 50.
9. You may use the attached Operator Precedence chart and Syntax chart
10. You will have 30 minutes to complete this Quiz.
11. When you are finished, bring your paper and student card to the front of the room where you will hand in your quiz.

Good luck!

Total = _____ / 50

Question	Max Mark	Actual Mark
1	5	
2	5	
3	10	
4	10	
5	10	
6	10	
Total	50	

1. Answer the following questions about the tree below. 5 Marks

- a. Identify the Root of this tree:
- b. List the Ancestors of Node E:
- c. What is the Height of the Tree?
- d. Is the tree Complete?
- e. What is the Order of this tree?

2. Complexity**5 Marks**

What is the complexity of the following code fragment using Big-O notation with respect to the value of n?

```
for (int count1 = 0; count1 < n; count1++)  
    for (int count2 = 0; count2 < 2*n; count2++)  
        cout << count2*count2*count2 << endl;
```

- ☐ $n + 2n$
- ☐ $O(n + 2n)$
- ☐ $O(n^3)$
- ☐ $O(n^2)$
- ☐ $O(1)$
- ☐ None of the above

3. Recursive Functions - fibonacci

10 Marks

The fibonacci sequence is a famous bit of mathematics, and it happens to have a recursive definition. The first two values in the sequence are 0 and 1 (essentially 2 base cases). Each subsequent value is the sum of the previous two values, so the whole sequence is: 0, 1, 1, 2, 3, 5, 8, 13, 21 and so on. Define a recursive fibonacci(n) method that returns the nth fibonacci number, with n=0 representing the start of the sequence.

Your function must have the following signature:

```
int fibonacci(int n);
```

For example:

```
fibonacci(0) → 0
fibonacci(1) → 1
fibonacci(2) → 1
fibonacci(3) → 2
fibonacci(4) → 3
```

```
int fibonacci(int n) {

}
}
```

4. Dynamic Arrays as Return Values - tripleChar**10 Marks**

Given a C string parameter called `str`, return a C string where for every char in the original, there are three chars. Your function will accept a nullptr or a C string that is terminated by a null (zero) character, and must return a new dynamic char array that is also terminated by a null character, and that contains three characters for every character in the `str` tripleChar.

Your function must have the following signature:

```
char* tripleChar(const char str[]);
```

For example:

<code>tripleChar("The")</code>	→ <code>"TTThhheee"</code>
<code>tripleChar("AAbb")</code>	→ <code>"AAAAAAbbbb"</code>
<code>tripleChar("Hi-There")</code>	→ <code>"HHHiii---TTThhheerrreee"</code>
<code>tripleChar(nullptr)</code>	→ <code>" "</code>

```
char* tripleChar(const char str[]) {
```

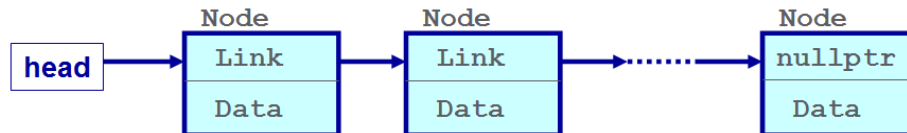
```
}
```

5. Implement Linked List functions below using `struct Node` 10 Marks

```

struct Node {
    Node    *link;
    int     data;
};
typedef Node* NodePtr;
NodePtr head = nullptr;

```

**(a) Complete the `head_insert` routine below:****5 Marks**

```

/*****
 * Creates a new Node, sets its data field to the_number, and adds it as the
 * first node pointed to be head.
 *****/
void head_insert(NodePtr& head, int the_number) {

}

```

(b) Complete the `search` routine below:**5 Marks**

```

/*****
 * This function searches the linked list for a node with data == dataToFind
 * and returns a pointer to the node if found, and returns nullptr otherwise.
 *****/
NodePtr search(NodePtr head, int dataToFind) {

}

```

6. Polymorphism: What output does main() produce?**15 Marks**

```
class Animal {
public:
    Animal(string animalType, string sound);
    virtual void speak();
    string animalType;
    string sound;
};

Animal::Animal(string animalType, string sound) {
    this->animalType = animalType;
    this->sound      = sound;
}

class Bee      : public Animal{ public: Bee (); };
class Lion     : public Animal{ public: Lion(); void speak() override;};
class MtnLion: public Lion   { public:      void speak() override;};

void Animal ::speak(){cout <<animalType <<"s say " << sound << endl;}
void Lion   ::speak(){cout <<"Big Cats ROAR!"           << endl;}
void MtnLion::speak(){cout <<animalType <<"s purr"       << endl;}

Bee ::Bee() : Animal("Bee", "Buzz Buzz") {}
Lion::Lion(): Animal("Lion", "Meow"      ) {}

int main(int argc, char** argv) {
    Animal bear("Bear", "Grrrr!");
    Animal* zoo[4] = {&bear, new Bee, new Lion, new MtnLion};
    Animal lion   = *zoo[3];

    cout << "Zookeeper, Zookeeper, What do you hear?\n";
    for(Animal* animal:zoo ) animal->speak();
    lion.speak();
}
```

(write the output produced by executing main here)

C++ Operator Precedence - Appendix 2

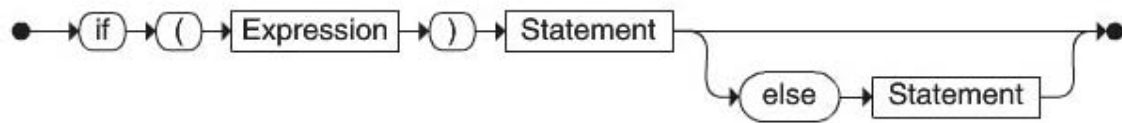
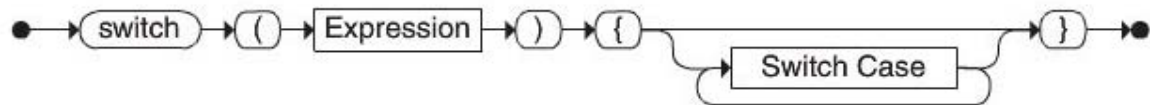
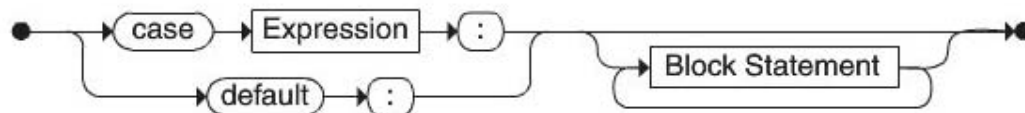
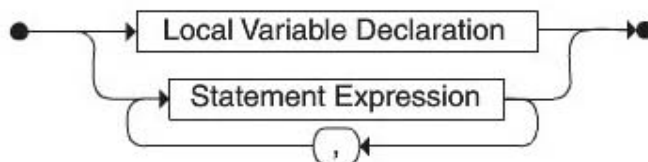
:: scope resolution operator
. dot operator -> member selection [] array indexing () function call ++ postfix increment operator (placed after the variable) -- postfix decrement operator (placed after the variable)
++ prefix increment operator (placed before the variable) -- prefix decrement operator (placed before the variable) ! not - unary minus + unary plus * dereference & address of new delete delete[] sizeof
* multiplication / division % remainder (modulo)
+ addition - subtraction
<< insertion operator (output) >> extraction operator (input)
< less than <= less than or equal > greater than >= greater than or equal
== equal != not equal
&& and
or
= assignment += add and assign -= subtract and assign *= multiply and assign /= divide and assign %= modulo and assign

*Highest precedence
(done first)**Lowest precedence
(done last)***C++ Flow of Control Statement Syntax**

Instructor: Scott Kristjanson

Wk14

TA: Wenqiang Peng

If Statement**Switch Statement****Switch Case****While Statement****For Statement****For Init****For Update**