I. Fill in the Blank (20 pts)

1. (2 pts) A destructor is a special member function that is called prior to an object being deallocated.

2. (2 pts) A dangling pointer can occur when we have two pointers which refer to the same object and we call delete on one of them.

3. (2 pts) An outside function can be defined to be a friend of the class definition and will be able to access the private members of the class definition.

4. (2 pts) A mutator function is a public function in a class definition which is used to change the value of private variables in the class definition.

5. (2 pts) The freestore is a special area of memory where dynamic variables are stored when they are created and we must be sure to return the space taken up by the variables so we don’t run out.

6. (2 pts) If we place a & in front of a variable, we can obtain the memory address of the variable.

7. (2 pts) The constructor is to initialize the member variables contained in the class definition.

8. (2 pts) The purpose of a copy constructor is to initialize the current object with the contents of the object that is sent as the parameter.

9. (2 pts) Encapsulation is a binding together of data and the code which operates on the data.

10. (2 pts) The class that is created by adding functionality to an existing class is called a derived class.
II. Short Answer (60 pts)

11. (4 pts) Suppose p is a pointer to a double and a double is stored in 8 bytes. If the value of p is 0200, what is the value of p+5?

\[02040\]

12. (4 pts) What are three requirements of an ADT?

1. All member variables are private.
2. All operations that a programmer needs to use the class should be public functions and there should be a full specification of how to use each member function.
3. Any helping functions are private. destructor

13. (4 pts) What is printed by the following loop?

\[
\begin{align*}
\text{int counter} & \text{ = 0;} \\
\text{while (++counter < 10)} \\
& \text{cout << counter << endl;}
\end{align*}
\]

Recall that the preincrement operator returns the value of the variable after the increment.

1 2 3 4 5 6 7 8 9

14. (4 pts) Suppose we have a class definition named Number. We wish to overload the = operator in the class definition. Show the declaration of the = operator in the class definition.

\[
\text{void operator=}\left(\text{const Number}& \text{ input}\right);
\]

15. (4 pts) When will a default constructor be generated for us and in which situations will it not?

A default constructor will be generated for us if we don’t define any constructors. If we define at least one constructor, then no default constructor will be generated.

16. (4 pts) What can we not do if we overload the = operator?

We cannot make it a friend of the class.

17. (4 pts) Suppose p is a pointer to an int. Show how you would create a dynamic int using p and assign 3 to it.

\[
\begin{align*}
\text{int}\ast & \text{ p = new int;} \\
& \text{*p = 3;}
\end{align*}
\]

18. (4 pts) How do we return space to the freestore?

Using delete
19. (4 pts) Suppose we want to define a class definition named Word with a char pointer named characters and an int named size as its member variables. Show the declaration of the constructor in the class definition that will accept one int parameter.

\[ \text{Word(int input)} \]

20. (4 pts) Suppose we have a class definition named TernaryNumber. Show how we would declare the destructor for this class in the class definition and how we would write the destructor in the implementation. You may leave the body empty.

\[ \text{TernaryNumber();} \]
\[ \text{TernaryNumber:: TernaryNumber() \{} \]
\[ \text{\}} \]

21. (4 pts) If we have a class definition called Vector, what is the syntax of the copy constructor in the class definition?

\[ \text{Vector(const Vector& input);} \]

22. (4 pts) Show the syntax for creating an array of ints called numbers with 5 spaces and then show a for loop which will initialize each element to 1.

\[ \text{int numbers[5];} \]
\[ \text{for (int counter=0; counter<5; counter++)} \]
\[ \text{numbers[counter] = 1;} \]

23. (4 pts) What is printed by the following?

\[ \text{int num = 1;} \]
\[ \text{int* p = \&num;} \]
\[ \text{*p = *p + 2;} \]
\[ \text{cout << num << endl;} \]

Recall that we obtain the address of the variable num with \&. We can then assign the address to a pointer variable. We can then use the dereferencing operator to affect the value of the variable. So since we change the value associated with the pointer, this changes the value of the variable. So 3 is printed.

24. (4 pts) What are the two ways variables can be sent to functions?

by value and by reference

25. (4 pts) Suppose we want to have a structure called RationalNumber which contains two member variables, numerator and denominator. Show the definition of the structure and how you would create an instance of the structure.

\[ \text{struct RationalNumber \{} \]
\[ \text{\quad int numerator;} \]
\[ \text{\quad int denominator;} \]
\[ \text{\};} \]
\[ \text{RationalNumber rationalNumber = \{1, 2\};} \]
III. Problems (20 pts)

26. (10 pts) Show the class definition for a Vector. The class should have one pointer variable of type int named p, one int named size, and one int named index which will indicate where the next number added to the Vector will go. Include one constructor which will accept one int parameter to initialize the variable size and then create a dynamic array of that size. Include one member function named add which will accept an int and place that int into the dynamic array. If the array is full, increase its size by 10 before adding the number to it. Show both the declaration and the implementation.

```cpp
#include <iostream>
using namespace std;

class Vector {
public:
    Vector(int input);
    void add(int number);

private:
    int* p;
    int size;
    int index;
};

int main() {
    return(0);
}

Vector::Vector(int input) {
    index = 0;
    size = input;
    p = new int[size];
}

void Vector::add(int number) {
    if (index == size) {
        int* p1 = new int[size+10];
        for (int counter=0;counter<size;counter++)
            p1[counter] = p[counter];
        delete(p);
        p = p1;
    }
    p[index++] = number;
}
```
27. (10 pts) Show the class definition for a ComplexNumber. The class should have two member variables: `realPart` and `imaginaryPart` both of type `double`. You should include one constructor that will accept two `doubles` and initialize `realPart` and `imaginaryPart`. You should also overload the `+` operator to find the sum of two `ComplexNumbers`. Show both the declaration of the class and the implementation.

```cpp
#include <iostream>
using namespace std;

class ComplexNumber {
public:
    ComplexNumber(double real, double imaginary);
    friend ComplexNumber operator +(const ComplexNumber& c1, const ComplexNumber& c2);
private:
    double realPart;
    double imaginaryPart;
};

int main() {
    return(0);
}

ComplexNumber::ComplexNumber(double real, double imaginary) {
    realPart = real;
    imaginaryPart = imaginary;
}

ComplexNumber operator +(const ComplexNumber& c1, const ComplexNumber& c2) {
    ComplexNumber sum = ComplexNumber(0, 0);
    sum.realPart = c1.realPart + c2.realPart;
    sum.imaginaryPart = c1.imaginaryPart + c2.imaginaryPart;
    return(sum);
}
```