CS20A Quiz 1

NASHIR JANMOHAMED

TOTAL POINTS

25 / 25

QUESTION 1

1 True/False 7 / 7

- **1 pts** 1. True
- **1 pts** 2. True
- **1 pts** 3. False
- **1 pts** 4. False
- **1 pts** 5. True
- **1 pts** 6. False
- **1 pts** 7. False

✓ **- 0 pts Correct**

QUESTION 2

2 Multiple choice 6 / 6

✓ **- 0 pts Correct**

- **1 pts** 8. d
- **1 pts** 9. d
- **1 pts** 10. a
- **1 pts** 11. b
- **1 pts** 12.c
- **1 pts** 13.b

QUESTION 3

3 Pointing Pointers 4 / 4

✓ **- 0 pts Correct**

- **0.5 pts** a. You have the idea, but there are syntax errors.
- **1 pts** a. incorrect
- **0 pts** ----------
- **0.5 pts** b. There is a hint of correctness.
- **0.5 pts** b. Syntax
- **1 pts** b. incorrect.
- **0 pts** ----------
- **0.5 pts** c. Syntax
- **0.5 pts** c. There is a hint of correctness.
- **1 pts** c. incorrect
- **1 pts** c. incomplete
- **0 pts** ----------
- **0.5 pts** d. -8
- **1 pts** d. incorrect
- **1 pts** d. incomplete
- **4 pts** Incomplete

QUESTION 4

4 Pointer Syntax 3 / 3

✓ **- 0 pts Correct**

- **1 pts** Logical Error
- **1 pts** Loop printing error
- **1 pts** *ptr inside loop
- **0 pts** ----------
- **3 pts** Incomplete
- **3 pts** Incorrect
- **0.5 pts** Introduced error

QUESTION 5

5 CountingPiggy 5 / 5

✓ **- 0 pts Correct**

- **0.5 pts** No Constructor Implementation
- **0.5 pts** Scope Resolution ::
- **1 pts** CoutingPiggy::giveMeBills incorrect
- **1.5 pts** CoutingPiggy::giveMeBills incomplete
- **1 pts** Major syntax
- **0.5 pts** Output Formatting
- **1 pts** Major Logic Error
- **0.5 pts** Minor syntax
- **0.5 pts** Minor Logic Error
- **5 pts** Incomplete

Santa Monica College CS 20A: Data Structures with C++ Quiz 1
Spring 2018

Name: Nashir Jammohamed **ID:** 1325605

True/False: Write out either True or False.

1. True If you do not define one the compiler will create an implicit constructor for your class.
2. True You can define more than one constructor for a class.
3. False When defining an instance of a class, member variables in that class are always initialized to known values.
4. False You can define more than one destructor for a class.
5. True Pointers in C++ are data types that store addresses.
6. False In C++ arrays are just pointers. No but they decay to pointers -> arrays are arrays
7. False Suppose you have an int pointer called ptr pointing to an integer array. C++ understands *(ptr) + 2 to mean "move two integers down and retrieve that value". *(ptr+2)

Multiple Choice:

8. d Abstract Data Types consist of:

   a.) Data Structures.
   b.) Algorithms.
   c.) An Interface.
   d.) All mentioned.

9. d Which of these best describes the concept of 'Encapsulation'?

   a.) Hides data and algorithms.
   b.) Using single interface for general class of actions.
   c.) Reduce Complexity.
   d.) All mentioned.

10. a If a class's data members are private, what can we do to access them from the class object? ambiguous?

    a.) Create public member functions to access those data members. ✓
    b.) Create private member functions to access those data members. ✗
    c.) Create protected member functions to access those data members.
    d.) Private data members can never be accessed from outside the class.

11. b What is the difference between struct and class in C++?

    a.) All members of a structure are public and structures don't have constructors and destructors.
    b.) Members of a class are private by default and members of struct are public by default. When deriving a struct from a class/struct, default access-specifier for a base class/struct is public and when deriving a class, default access specifier is private.
    c.) All members of a structure are public and structures don't have virtual functions.
    d.) All mentioned.

12. C What is the output of the following program?

```
#include<iostream>
using namespace std;
class Point {
public:
    Point() { cout << "Constructor called" << endl; }
};

int main() {
   Point t1, *t2;
   return 0;
}
```

constructor not called

a.) Compiler Error.
b.) Constructor called
Constructor called
c.) Constructor called
d.) Nothing will print.

13. b You must include a classes header file when.

a.) Use the class as a parameter to a function. ✗
b.) Use any of that classes member functions. ✓
c.) Declare a pointer or reference to that class. ✗
d.) All mentioned.

Short Answer:

14. For this problem, you will be asked to write some code to accomplish a particular task given the code fragment below. Each task will depend on the tasks that came before it. Your code must be syntactically correct.

```
int i1 = 100;
int i2 = -8;
int i3 = 15;

int *p1, **p2, ***p3;
```

a.) Set p1 to point to i1.

```
p1 = &i1;
```

b.) Using p2 change the value of i1 to the value of i2.

```
p2 = &p1;  **p2 = i2;
```

c.) Using p3 make p1 point to i3

```
p3 = &p2;  **p3 = &i3;
```

d.) What does the following code output? cout<< &*&*&*p2; //p2 points to p1 → p1 points to i3

15

15. This following program is supposed print 30 20 10, but it does not. Find all the bugs and show fixed corrected version of the program. Note that cout<<"30 20 10"; is not a correct fix. if only

```
int main(){
      int arr[3] = { 5, 10, 15 };
      int* ptr = arr; //ptr = &arr

      *ptr = 10;            // set arr[0] to 10 ✓
      *(ptr + 1) = 20;      // set arr[1] to 20 ✗
      ptr += 2;             // ptr points to arr[2] ✓
      ptr[0] = 30;          // set arr[2] to 30 ✓

      while (ptr >= arr) {
            ptr--;          //if you decrement first, you start at 2nd element (20)
            cout << " " <<*ptr;    // print values
      }     ptr--;
      cout << endl;
}
```

need double quotes?

```
int main(){
    int arr[3] = {5, 10, 15};
    int* ptr = &arr; //just to be safe!

    *ptr = 10; //points to arr[0], sets to 10

    *(ptr+1) = 20; //points to arr[1], but doesn't change ptr, sets to 20

    ptr += 2; //points to arr[2]

    ptr[0] = 30; //"                    ", sets to 30

    while (ptr >= &arr) { //just being explicit with &
          cout << " " << *ptr;
          ptr--;
    }
    cout << endl;
    return 0;
}
```

16. Suppose you're tasked with implementing a coin counter program for a digital piggy bank. The incomplete class declaration is as follows:

```
class CountingPiggy
{
public:
      CountingPiggy ();
      void addDollar();      // Add a dollar coin to the machine.
      void addQuarter();     // Add a quarter to the machine.
      void addDime();        // Add a dime to the machine.
      void addNickel();      // Add a nickel to the machine.
      void addPenny();       // Add a penny to the machine.
      void giveMeBills();    // Print what the user will get.
                             // See the example below.
private:
// TODO: Add private variables here.
      int m_pennies;


};
```

The class's usage may look like:

```
CountingPiggy  cp;
cp.addDollar();                  // Currently $1.00.
cp.addQuarter();                 // Currently $1.25.
for (int   i = 0; i < 10;   i++) // Add 10 dimes, total becomes $2.25.
      cp.addDime();
for (int   i = 0; i < 32;   i++) // Add 32 pennies, total becomes $2.57.
      cp.addPenny();
cp.giveMeBills();      // Produces the following output.
```

```
Output:
2 dollar bill(s)
2 quarter(s)
0 dime(s)
1 nickel(s)
2 penny(ies)
```

Implement all the member functions so that the machine works as intended. Do not add or modify any of the public members. You are free to add any private members as needed. It may be useful to recall that in C++ the modulus operator is % which gives you the remainder in a division operation.

// store dollars as 100 x pennies

```
#include "CountingPiggy.h"
#include <iostream>
CountingPiggy::CountingPiggy(){m_pennies=0;}
void CountingPiggy::addDollar(){m_pennies+=100;}
void CountingPiggy::addQuarter(){m_pennies+=25;}
void CountingPiggy::addDime(){m_pennies+=10;}
void CountingPiggy::addNickel(){m_pennies+=5;}
void CountingPiggy::addPenny(){m_pennies+=1;}
void CountingPiggy::giveMeBills(){
    int dollars=0, quarters=0, dimes=0, nickels=0, pennies=0;
    while((m_pennies-100)>0){
        dollars++;
        m_pennies-=100;
    }while((m_pennies-25)>0){
        quarters++;
        m_pennies-=25;
    }while((m_pennies-10)>0){
        dimes++;
        m_pennies-=10;
    }while((m_pennies-5)>0){
        nickels++;
        m_pennies-=5;
    }while((m_pennies-1)>0){
        pennies++;
        m_pennies-=1;
    }
    std::cout<<dollars<<" dollar bill(s)"<<std::endl;
    std::cout<<quarters<<" quarter(s)"<<std::endl;
    std::cout<<dimes<<" dime(s)"<<std::endl;
    std::cout<<nickels<<" nickel(s)"<<std::endl;
    std::cout<<pennies<<" penny(ies)"<<std::endl;
}
```

```
//2.57
//2 dollars
//2 quarters
//1 nickels
//2 pennies
```