Answer:

Final Exam F24 Early Final
CSCI 13500: Software Analysis and Design 1
Hunter College, City University of New York

Dec 16, 2024, 1:45 PM - 3:45 PM, North Building Lab 1001B

Exam Rules

- Show all your work. Your grade will be based on the work shown.
- The exam is closed book and closed notes with the exception of a provided cheat sheet.
- When taking the exam, you may bring pens and pencils.
- Scratch paper is provided. For your convenience, you may take the scratch paper and cheat sheet off. But make sure **not** to put solutions to the scratch paper.
- You may not use a computer, calculator, tablet, phone, earbuds, or other electronic device.
- **Do not open this exam until instructed to do so.**

Hunter College regards acts of academic dishonesty (e.g., plagiarism, cheating on examinations, obtaining unfair advantage, and falsification of records and official documents) as serious offenses against the values of intellectual honesty. The College is committed to enforcing the CUNY Policy on Academic Integrity and will pursue cases of academic dishonesty according to the Hunter College Academic Integrity Procedures.

I understand that all cases of academic dishonesty will be reported to the Dean of Students and will result in sanctions.									
Name:									
EmpID:									
Email:									
Signature:									

1 (30 points) Answer the following questions.

(1) Given `string groceries[] = {"milk", "apple pie", "onion"}`, what is groceries[1].substr(3, 5)?

Answer: groceries[1].substr(3, 5) is "le pi". Explanation: groceries[1] is the second element of array of strings, which is "apple pie". Expression groceries[1].substr(3, 5) is the substring from the fourth letter of this string spanning with 5 letters, which is substring "le pi".

(2) Given a declaration `std::vector<int> v(3, 1);`, what is the value of v.size()?

Answer: `v.size()` returns the number of elements of v, which is 3 in this example.

(3) What possible numbers does code `1 + rand() % 7` generate?

Answer: Answer: rand() % 7 generate a random int in [0, 6].

`1 + rand() % 7` generates a random int in [1, 7].

(4) Given `string numStr = std::to_string(-21) + "0";`, where `to_string` converts an integer to a string. What is the value for numStr?

Answer: the answer is "-210".

(5) What is the value of `(4 + 7 % 3) / 2` in C++?

Answer: 2

Explanation: division operator % has higher precedence than addition operator +. So % runs first in `4 + 7 % 3`. Note that 7 % 3 is 1, after it adds with 4, the sum is 5. When 5 is divided by 2, the quotient is 2.

(6) Write **header** of a function called min, given an array of characters (type char) with *size* many elements, return the smallest ASCII code of all the elements in the array.

Answer: `int min(char* arr, int size);` or `int min(char arr[], int size);`

(7) Declare class Coord as follows.

```
1 class Coord {
2 public:
3     double x;
4     double y;
5 };
```

Declare a Coord object `point` and initialize its x as 2 and y as 3.

Answer:

```
1 Coord point = {2, 3};
```

or

```
Coord point{2, 3};
```

or

```
Coord point;
point.x = 2;
point.y = 3;
```

(8) Given `int grades[] = {73, 99, 100, 62};` What is the value of `*grades + 2`?

Answer: 75

(9) Given the following code segment.

```
void foo(string *ps, string *pt);

int main() {
    string s = "hello";
    string t = "hi";

    //TODO: write a statement to call foo using appropriate attributes of s and t.

    return 0;
}
```

Answer: `foo(&s, &t)`

(10) Suppose we have main function defined as follows. And calling `foo(m, n)`, the values of m and n are swapped. That is, m becomes 2 and n becomes 1.

```
int main() {
   int m = 1;
   int n = 2;
   foo(m, n);
   return 0;
}
```

What is the **header** of function foo? Suppose its return type is void.

Answer: `void foo(int& m, int& n);` //note that `&` after int cannot be omitted and it means pass by reference.

(11) What is output for the following code?

```
    int a = 2;
    int* p = &a;
    a++;
    cout << *p << endl;
```

Answer: 3

Explanation: after `int* p = &a`, which saves a's address to pointer p, then $*p$ represents the guy who lives in the address of variable a. Note that no two variables can reside in the same address, so *p is an alias of variable a.

a++; is the same as $a = a + 1$;; so a changes from the initital value 2 to 3. Then *p is 3.

(12) What is the output for the following code?

```
    vector<int> nums = {1, 2, 0};

    int count = 0;
    for (int i = 0; i < nums.size(); i++)
        if (nums[i] % 2 == 0)
           count++;

    cout << count << endl;
```

Answer: 2

(13) What the output of the following code?

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

int main() {
    for (int row = 0; row < 3; row++) {
        for (int col = 0; col < 2; col++) {
            if (row >= 1)
               cout << "#";
            else cout << "-";
        }
        cout << endl;
    }
    return 0;
}
```

Answer:

```
--
##
##
```

(14) What is panel after slide up operation?

1	6	3
2		7
4	5	8

Answer:

1	6	3
2	5	7
4		8

(15) Suppose in Project 3, data member `bins` have the following values,

`{{2, 1, 3}, {1, 1, 3}, {2, 2}, {3}}`,

After moving eligible element(s), according to rules listed in Project 3, from the second to the left bin to the rightmost bin, what are the elements in the rightmost bin?

Answer: 3, 3

2 (15 points) Answer the following questions.

(1) Define function `countSuccessiveBackElms`, for an given array of integers with its size, return the number of successive elements in the end of this array.

For example, call the function with array with values 1, 0, -1, 0, 0, the size of array is 5. There are two zeros residing successively (aka consecutively) on the end of array, the return is 2. Note that the second zero to the left is not adjacent with other zeros at the end, so it is not counted as part of the results.

In main function, write the following statements. No need to write the full definition of main function.

Define an int array with elements 1, 0, -1, 0, 0.

Call and print the number of successive back elements of the above array.

Answer:

```
int countSuccessiveEndElms(int* arr, int size) {
   int i = size-1;
   int elm = arr[i];
   int count = 0;
   while (i >= 0 && arr[i] == elm) {
       count++;
       i--;
   }

   return count;
}
```

A complete code to define and test the above function is as follows.

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

int countSuccessiveEndElms(int* arr, int size);

int main() {
    int arr[] = {1, 0, -1, 0, 0};
    int size = sizeof(arr) / sizeof(arr[0]);

    cout << countSuccessiveEndElms(arr, size) << endl;
    return 0;
}

int countSuccessiveEndElms(int* arr, int size) {
   int i = size-1;
   int elm = arr[i];
   int count = 0;
```

```
    while (i >= 0 && arr[i] == elm) {
        count++;
        i--;
    }

    return count;
}
```

(2) Write a function, given an array of integers, its size, and a target (an integer), return a pointer to the first occurrence of the target in an array, or nullptr if there is no match.

For example, suppose an array has elements 1, 2, 3, 2, if the target is 2, then the return of the function is a pointer to the second element to the left. if the target is 4, then the return is nullptr.

Answer:

```
int* first_occurrence(int arr[], int size, int target) {
    for (int i = 0; i < size; i++)
        if (target == arr[i])
          return arr + i;

    return nullptr;
}
```

A complete code is as follows.

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

int* first_occurrence(int arr[], int size, int target);

int main() {
    int arr[] = {1, 2, 3, 2};
    int size = sizeof(arr) / sizeof(arr[0]);

    int* p = first_occurrence(arr, size, 2);

    cout << p << endl; //value depends on system and running time
    cout << *(p+1) << endl; //3

    int *p2 = first_occurrence(arr, size, 4);
    cout << p2 << endl;

    return 0;
}

int* first_occurrence(int arr[], int size, int target) {
    for (int i = 0; i < size; i++)
        if (target == arr[i])
          return arr + i;

    return nullptr;
}
```

3 (10 points) Programming exercise on class

1. Define class for representing time in the 24-hour format. It is reasonable to define it to have two integer fields:

   h for the number of hours, and m for the number of minutes.

```
class Time {
public:
    int h;
    int m;
};
```

   Define Time minusMinutes(Time curr, int mins);

   The function should create and return a new moment of time that is mins minutes before curr. Example:

   `minusMinutes({8, 10}, 75) // should return {6, 55}`

   (We will not test how your function behaves if the new returned time will be on the previous day, feel free to assume that it will remain withing the same day, $\leq$ 23:59.)

2. In main function, do the following,

   - Define `curr` as a Time object with h equals 8 and m equals 10.
   - Call `minusMinutes` to find out and return a Time object that is 75 minutes before `curr`.

Answer:

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

class Time {
public:
    int h;
    int m;
};

Time minusMinutes(Time curr, int mins);

int main() {
    Time curr = {8, 10};
    Time prev = minusMinutes(curr, 75);
    cout << prev.h << ":" << prev.m << endl; //print 6:55
    return 0;
}

Time minusMinutes(Time curr, int mins) {
```

```
    int totalMinutes = curr.h * 60 + curr.m;
    totalMinutes -= mins;

    Time prev = {totalMinutes / 60, totalMinutes \% 60}; %TODO, when commenting \
       answertrue, need to add change % 60 to \% 60.

    return prev;
}
```

4 (10 points) Write codes of vector

Define a function called `choose`, for a vector `v` of characters (type char), return a vector with all the elements from `v` that are alphanumerical, in the same order. In English, alphanumeric characters are the 26 letters of the alphabet and the 10 Arabic numerals.

For example, given a vector of characters with elements `'a', 'b', '#', '1', '?'`, the return is a vector with elements `'a', 'b', '1'`.

Hint: int isalnum (int c); Check if character is alphanumeric. A value different from zero (i.e., true) if indeed c is either a digit or a letter. Zero (i.e., false) otherwise.

isalnum is from cctype library.

Answer:

```
vector<char> choose(vector<char> v) {
    vector<char> results;
    for (int i = 0; i < v.size(); i++) {
        if (isalnum(v[i]))
           results.push_back(v[i]);
    }

    return results;
}
```

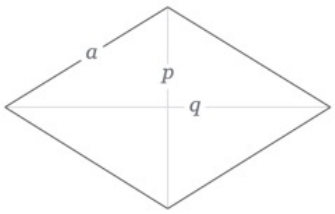
A complete code is shown as follows.

```
#include <iostream>
#include <string>
#include <cctype>
#include <vector>
using namespace std;

vector<char> choose(vector<char> v);

int main() {
    vector<char> v = {'a', 'b', '#', '1', '?'};

    vector<char> results = choose(v);

    for (int i = 0; i < results.size(); i++)
        cout << results[i] << " ";

    cout << endl;

    return 0;
}

vector<char> choose(vector<char> v) {
    vector<char> results;
```

```
    for (int i = 0; i < v.size(); i++) {
        if (isalnum(v[i]))
           results.push_back(v[i]);
    }

    return results;
}
```

5 (15 points) Define class for diamond shape.

1. In a diamond shape, also called rhombus, all the sides are equal in length, and the diagonals bisect each other at right angles. It has two parameters:



   (a) side **a**

   (b) a short diagonal **p**

2. **Assume that Diamond.hpp is provided** where data members **a** and **p** are declared as double types. **Your job** is to define Diamond.cpp with the following requirement.

3. Define a default constructor, set data members a to be 1 and p to be 1.

   Answer:

```
Diamond::Diamond() {
    a = 1;
    p = 1;
}
```

4. Define a non-default constructor, which takes formal parameters a and p, both are double types.

   (a) If both a and p are positive and 2a > p, set data member **a** by given parameter a and set data member **p** by given parameter p.

   (b) otherwise, set data members **a** and **p** to be 1.

   Answer:

```
Diamond::Diamond(double a, double p) {
    if (a > 0 && p > 0 && 2 * a > p) {
       this->a = a;
       this->p = p;
    }
    else {
        this->a = 1;
        this->p = 1;
    }
}
```

5. Define method **getSide**, return the value of a.

Answer:

```
double Diamond::getSide() const {
    return a;
}
```

6. Define method **getArea**, which defined by $\frac{1}{2}p\sqrt{4a^2 - p^2}$, where `sqrt` is a function from `cmath` library.

Answer:

```
double Diamond::getArea() const {
    return 1.0 / 2 * p * sqrt(4*a*a - p*p);
}
```

7. Define method **getPerimeter**, which returns the product of 4 times a.

Answer:

```
double Diamond::getPerimeter() const {
    return 4*a;
}
```

Define **DiamondTest.cpp**, do the following:

1. Create an Diamond object named **dia** from its default constructor.

Answer:

```
    Diamond dia;
```

2. Find out and print the area of **dia**.

Answer:

```
    cout << "area of diamond: " << dia.getArea() << endl;
```

3. Find out and print the perimeter of **dia**.

Answer:

```
    cout << "perimeter of diamond: " << dia.getPerimeter() << endl;
```

Answer: A complete code is as follows.
codes of Diamond.hpp

```
#ifndef DIAMOND_H
#define DIAMOND_H

class Diamond {
public:
    Diamond();
    Diamond(double a, double p);
    double getArea() const;
    double getPerimeter() const;
    double getSide() const;
    void setSide(double side);
private:
    double a;
    double p;
};
#endif
```

codes of Diamond.cpp

```
#include "Diamond.hpp"
#include <cmath> //sqrt

Diamond::Diamond() {
    a = 1;
    p = 1;
}

Diamond::Diamond(double a, double p) {
    if (a > 0 && p > 0 && 2 * a > p) {
       this->a = a;
       this->p = p;
    }
    else {
        this->a = 1;
        this->p = 1;
    }
}

double Diamond::getSide() const {
    return a;
}

double Diamond::getPerimeter() const {
    return 4*a;
}

```

```
double Diamond::getArea() const {
    return 1.0 / 2 * p * sqrt(4*a*a - p*p);
}

void Diamond::setSide(double side) {
    if (side > 0 && 2*side > p)
       a = side;
}
```

contents of DiamondTest.cpp

```
#include <iostream>
#include <string>
#include "Diamond.hpp"
using namespace std;

//sample output:
//area of diamond: 1.51987
//perimeter of diamond: 6.4
int main() {
    Diamond dia;
    dia.setSide(1.6);
    cout << "area of diamond: " << dia.getArea() << endl;
    cout << "perimeter of diamond: " << dia.getPerimeter() << endl;
    return 0;
}
```

6 (10 points) function on vectors

Given two vectors of integers, if they have the same number of elements, find out whether every element in the first is smaller than the corresponding element in the second vector, if yes, return true, otherwise, return false. If these vectors do not have the same number of elements, return false.

For example, if the first vector is {1, 2, 3} and the second vector is {2, 3, 6}, return true.

If the first vector is {1, 2} and the second vector is {2, 3, 4}, the return is false.

Answer:

```
#include <iostream>
#include <string>
#include <vector>
using namespace std;

//Given two vectors of integers, if they have the same number of elements, find out whether every element
//in the first is smaller than the corresponding element in the second vector, if yes, return true, otherwise,
//return false. If these vectors do not have the same number of elements, return false.
//For example, if the first vector is {1, 2, 3}and the second vector is {2, 3, 6}, return true.
//If the first vector is {1, 2}and the second vector is {2, 3, 4}, the return if false.

//bool compare(vector<int> v1, vector<int> v2);
bool compare(const vector<int>& v1, const vector<int>& v2);

int main() {
    vector<int> v1 = {1, 2, 3};
    vector<int> v2 = {2, 3, 6};

    cout << boolalpha << compare(v1, v2) << endl; //true

    vector<int> v3 = {1, 2};
    vector<int> v4 = {2, 3, 6};

    cout << boolalpha << compare(v3, v4) << endl; //false

    return 0;
}

bool compare(const vector<int>& v1, const vector<int>& v2) {
    if (v1.size() != v2.size())
       return false;

```

```
33      //v1.size() == v2.size()
34      for (int i = 0; i < v1.size(); i++) {
35          if (v1[i] >= v2[i])
36              return false;
37      }
38
39      return true;
40  }
```

7 (10 points) Define recursive function

Define a recursive function `printRangeRev`, given two integers left and right, representing the left and right boundary of a range. Print all numbers in range left $\leq$ x $\leq$ right in reverse order, separated by a space, in the same line. (Don't use loops, global or static variables.)

For example, if left is 1 and right is 3, the print is

3 2 1

Hint: what if left > right?

Warning: If you do not use recursion, you will not get any point. No repetition statement is allowed in this function.

Answer:

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

void printRangeRev(int left, int right);

int main() {
    printRangeRev(1, 3);
    return 0;
}

void printRangeRev(int left, int right) {
    if (left > right)
       return;

    if (left == right) {
       cout << left;
       return;
    }

    cout << right;
    cout << " ";
    printRangeRev(left, right-1);
}
```