Answer:

Final Exam S25 Final V2

CSCI 13500: Software Analysis and Design 1

Hunter College, City University of New York

May 21, 2025, 11:30 AM - 1:30 PM, N118

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.
- Unless the problem explicitly requests, no need to include libraries and using namespace std.
- **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[] = {"cake mix", "grape juice", "apple pie"}`, what is the value of groceries[0].substr(2, 5)?

Answer: groceries[0].substr(2, 5) is "ke mi". Explanation: groceries[0] is the first element of array of strings, which is "cake mix". Expression groceries[0].substr(2, 5) is the substring from the third letter – index 2 – of this string spanning with 5 letters, which is substring "ke mi".

(2) Given a declaration `std::vector<int> v(2, 1); v.push_back(0);`, what is the value of v.size()?

Answer: 3

explanation: `vector<int> v(2, 1);` creates a vector of integers with 2 elements, each element is 1. After push one more element to it, v has one more element. Hence, `v.size()` returns the number of elements of v, which is 3 in this example.

(3) What is the **minimum** integer that expression rand() % 7 -1 can generate?

Answer: -1

`rand() % 7` generates a random int in [0, 6].

`rand() % 7 - 1` generates a random int in [-1, 5].

(4) Given Given int `num = std::to_string(135).size() -2;`, where `to_string` converts an integer to a string and `size` method returns the number of characters of a string. What is the value for num?

Answer: 1. `std::to_string(135)` converts 135 to string "135", the size of string "135" is 3, and 3 - 2 is 1.

(5) What is the value of `8 / (1 + 2) % 3` in C++?

Answer: 2

Explanation: (1) expression in parentheses runs first. 1 + 2 = 3. (2) Run 8 / 3 and get 2. It is like to divide 8 pens among 3 persons, each person get 2 pens, two pen left. (3) 2 % 3 returns 2. Divide two pens among 3 persons, each person gets 0 pen, there are two pens left.

(6) Write **header** of a function called hasEmptyStr, given an array *arr* of string type with *size* many elements, return whether the array has at least an empty string or not. If yes, return true, otherwise, return false.

Answer: `bool hasEmptyStr(string* arr, int size);` or

`bool hasEmptyStr(string arr[], int size);`

(7) Declare class Time as follows.

```
class Time {
public:
    int hour;
    int minute;
};
```

Declare a Time object `curr` and initialize its hour as 8 and minute as 26.

Answer:

```
Time curr = {8, 26};
```

or

```
Time curr{8, 26};
```

or

```
Time curr;
curr.hour = 8;
curr.minute = 26;
```

(8) Given `int grades[] = {67, 92, 62};` What is the value of `*grades + 1`?

Answer: 68

(9) Given the following code segment.

```
//foo works with array pf of int type with size many elements
void foo(int *pf, int size);

int main() {
    int *arr = new int[20];

    //TODO: write a statement to call foo for dynamically allocated array arr and
       its size.
    //WRITE YOUR ANSWER IN THE FOLLOWING BOX.



    delete[] arr;
    arr = nullptr;

    return 0;
}
```

Answer: `foo(arr, 20);`

(10) Suppose we have main function defined as follows.

```
int main() {
   double a = 1.6;
   int b = foo(&a, 't');
   return 0;
}
```

What is the **header** of function foo?

Answer: `int foo(double* a, char ch);` or

`int foo(double*, char);`

(11) What is output for the following code?

```
string s = "12";
string *p = &s;
*p += "ab";
cout << s << endl;
```

Answer: 12ab

Explanation: after `string* 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.

*p += "ab"; is the same as $*p = *p +$ "ab";; which concatenate "ab" to the end of *p. Thus, $*p$ changes from the initital value "12" to "12ab". Then s is "12ab".

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

```
vector<int> nums = {-2, 0, -1, 2, -5};

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

cout << count << endl;
```

Answer: 3

Find out all negative integers in nums.

(13) What the output of the following code?

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

int main() {
    for (int row = 0; row < 4; row++) {
        for (int col = 0; col < 3; col++) {
            if (col % 2 != 0)
               cout << "*";
            else cout << "#";
        }
```

```
        cout << endl;
    }
    return 0;
}
```

Answer:

```
#*#
#*#
#*#
#*#
```

(14) What is the output of the following code? Assume that libraries and standard namespace are set up.

```
void foo(vector<string>& v);

int main() {
    vector<string> v = {"hey", "hi", "hello"};
    foo(v);

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

    return 0;
}

void foo(vector<string>& v) {
    int i = 0;
    int j = v.size() - 1;
    while (i < j) {
        swap(v[i], v[j]);
        i++;
        j--;
    }
}
```

Answer: ”hello hi hey ” (without double quotes) and followed by a return key.

(15) Given the following code, fill in the TODO part.

```
class Coord2D {
public:
    double x; //x-coordinate
    double y; //y-coordinate
};
```

```
6
7  double foo(Coor2D point) {
8      //TODO: return the product of x- and y-coordinates of point
9      //WRITE YOUR CODE IN THE FOLLOWING BOX.
10
11 }
```

Answer: return point.x * point.y;

A complete code is as follows.

```
1  #include <iostream>
2  #include <string>
3  using namespace std;
4
5  class Coord2D {
6  public:
7      double x;
8      double y;
9  };
10
11 double foo(Coord2D point);
12
13 int main() {
14     Coord2D point = {3, 5};
15     cout << foo(point) << endl;
16     return 0;
17 }
18
19 double foo(Coord2D point) {
20     return point.x * point.y;
21 }
```

2 (15 points) Answer the following questions.

(1) Define a function, `digit_space_only`, for a given string s, if it is **non-empty** and contains **only** digits and spaces, return true, otherwise, return false.

For example, `digit_space_only("")` returns false since it is an empty string.

`digit_space_only("12 3")` returns true.

`digit_space_only("12A b")` returns false since 'A' is not a digit or a space.

Hint: you may use the following functions from `cctype` library.

int isdigit (int c); Check if character is digit or not

int isspace (int c); Check if character is a whitespace or not

Answer:

```
//Define a function, digit_space_only, for a given string s,
//if it is non-empty and contains only digits and spaces, return true, otherwise,
   return false.
#include <iostream>
#include <string>
using namespace std;

bool digit_space_only(string s);

int main() {
    cout << boolalpha << digit_space_only("") << endl; //false
    cout << boolalpha << digit_space_only("123") << endl; //true
    cout << boolalpha << digit_space_only("123 456") << endl; //true
    cout << boolalpha << digit_space_only("123 A b") << endl; //false

    return 0;
}

bool digit_space_only(string s) {
    if (s == "")
       return false;

    for (int i = 0; i < s.size(); i++)
        if (!isdigit(s[i]) && !isspace(s[i]))
           return false;

    return true;
}
```

(2) Write a function `pointerToMin` that returns a **pointer** to the **first** appearance (if there are more than one occurrence) of the minimum value of an array of double type with *size* many elements.

If size is 0, return nullptr.

For example, suppose an array has elements 1.1, 3.3, 2.2, 3.3, 1.1, then the return of the function is a pointer to the first element.

Hint: you may use an index to the minimum element. Then use index and array name to get the pointer.

Answer:

```
double* pointerToMin(double* arr, int size) {
    if (size == 0)
       return nullptr;

    int minIdx = 0;
```

```
    for (int i = 0; i < size; i++)
        if (arr[i] < arr[minIdx])
          minIdx = i;

    return minIdx + arr;
}
```

A complete code is as follows.

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

double* pointerToMin(double* arr, int size);

int main() {
    double arr[] = {1.1, 3.3, 2.2, 3.3};
    int size = sizeof(arr) / sizeof(arr[0]);

    cout << pointerToMin(arr, size) << endl; //a pointer to the second element
    cout << pointerToMin(arr, size) - arr << endl; //0
         //show offset of minimum element to the initial address of array

    return 0;
}

double* pointerToMin(double* arr, int size) {
    if (size == 0)
       return nullptr;

    int minIdx = 0;
    for (int i = 0; i < size; i++)
        if (arr[i] < arr[minIdx])
          minIdx = i;

    return minIdx + arr;
}
```

3 (10 points) Programming exercise on class

1. Define class for representing length in feet and inches. It is reasonable to define it to have two integer fields:

   `foot` for the number of feet, and

   `inch` for the number of inches. Note that a foot has 12 inches, so we need to make sure that inch is in [0, 11].

Declare class `Length` with public data members `foot` and `inch`, both of `int` type.

Define non-member function `subtract`, given Length objects len and len2, the function should create and return a length object that is the result of subtracting `len2` from `len`. Example:

```
subtract({4, 6}, {2, 8}) // should return {1, 10}
```

Reason: 4 feet and 6 inches is 4 * 12 + 6 = 54 inches. And 2 feet 8 inches is 2 * 12 + 8 = 32 inches. Then 54 - 32 = 22 inches, which equals 1 feet and 10 inches.

Hint: For simplicity, we assume that `len` is no shorter than `len2`.

Answer:

```
class Length {
public:
    int foot;
    int inch; //value in [0, 11]
};
```

Answer:

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

class Length {
public:
    int foot;
    int inch; //value in [0, 11]
};

Length subtract(Length len, Length len2);

int main() {
    Length len = {4, 6};
    Length len2 = {2, 8};

    Length total = subtract(len, len2);
    cout << total.foot << " " << total.inch << endl; //1 10

    return 0;
}

Length subtract(Length len, Length len2) {
    int total_inches = len.foot * 12 + len.inch;
    int total_inches2 = len2.foot * 12 + len2.inch;

    int diff = total_inches - total_inches2;
```

```
    return {diff / 12, diff % 12};
}
```

4 (10 points) Write codes of vector

Define a function called `choose`, for a vector `v` of strings and a character (type char) `ch`, return a vector with all the elements from `v` whose strings **ending** with ch, in the same order. String `s` **ends** with character `ch` means `ch` is the **last** character of `s`.

For example, given a vector of strings with elements `"apple", "banana", "", "CDE", "orange"` and character 'e', the return is a vector with elements `"apple", "orange"`. Note that C++ is a case sensitive language, so 'e' is different from 'E'.

Hint: you may need to consider the case when a string is empty.

Answer:

```
vector<string> choose(vector<string> v, char ch) {
    vector<string> result;

    string s;
    int len;
    for (int i = 0; i < v.size(); i++) {
        s = v[i];
        len = s.length(); //length of string is the same as size of string
        if (len > 0 && s[len-1] == ch)
           //len > 0 means s, aka v[i], is not an empty string
           result.push_back(v[i]);
    }

    return result;
}
```

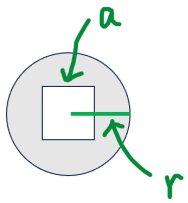
A complete code is shown as follows.

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

//sample output:
//apple
//orange
vector<string> choose(vector<string> v, char ch);

int main() {
    vector<string> v = {"apple", "", "banana", "CDE", "orange"};
    vector<string> result = choose(v, 'e');

```

```
    for (string elm: result)
        cout << elm << endl;

    return 0;
}

vector<string> choose(vector<string> v, char ch) {
    vector<string> result;

    string s;
    int len;
    for (int i = 0; i < v.size(); i++) {
        s = v[i];
        len = s.length(); //length of string is the same as size of string
        if (len > 0 && s[len-1] == ch)
           result.push_back(v[i]);
    }

    return result;
}
```

5 (15 points) Define class.

1. Define a SqCir as the region between a square nested into a circle. The shapes are concentric (share the same center). It has two parameters:



   (a) edge of the square **a**
   (b) radius of the circle **r**

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

3. Define a default constructor, set data members **a** to be 1 and **r** to be 2.

Answer:

```
SqCir::SqCir() {
    a = 1;
    r = 2;
}
```

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

   (a) If both a and r are positive and $\sqrt{2}$a is smaller than or equal to 2r, set data member **a** by given parameter a and set data member **r** by given parameter r.

   (b) otherwise, set data members **a** to be 1 and **r** to be 2.

Answer:

```
SqCir::SqCir(double a, double r) {
    if (r > 0 && a > 0 && sqrt(2) * a <= 2*r) {
       this->a = a;
       this->r = r;
    }
    else {
        this->a = 1;
        this->r = 2;
    }
}
```

5. Define method **getArea**, return the value of $\pi r^2 - a^2$, where π is defined as `M_PI` in `cmath` library. Note that a and r are data members.

Answer:

```
double SqCir::getArea() const {
    return M_PI * r * r - a * a;
}
```

6. Define method **getPerimeter**, which returns $4a + 2\pi r$. Note that a and r are data members.

Answer:

```
double SqCir::getPerimeter() const {
    return 4 * a + 2 * M_PI * r;
}
```

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

7. Create a SqCir object named **shape** from its non-default constructor with the edge of the square as 1 and the radius of the circle as 2.5.

Answer:

```
    SqCir shape(1, 2.5);
```

8. Find out and print the area of **shape**.

Answer:

```
cout << "area: " << shape.getArea() << endl;
```

9. Find out and print the perimeter of **shape**.

Answer:

```
cout << "perimeter: " << shape.getPerimeter() << endl;
```

Answer: A complete code is as follows.
code of SqCir.hpp

```
#ifndef SQ_CIR_H
#define SQ_CIR_H
class SqCir {
public:
    SqCir();
    SqCir(double a, double r);
    double getArea() const;
    double getPerimeter() const;

private:
    double a; //edge of the square
    double r; //radius of the circle
};
#endif
```

Code of SqCir.cpp

```
#include "SqCir.hpp"
#include <cmath>

SqCir::SqCir() {
    a = 1;
    r = 2;
}

SqCir::SqCir(double a, double r) {
    if (r > 0 && a > 0 && sqrt(2) * a <= 2 * r) {
       this->a = a;
       this->r = r;
    }
    else {
        this->a = 1;
        this->r = 2;
    }
}
```

```
19
20 double SqCir::getArea() const {
21     return M_PI * r * r - a * a;
22 }
23
24 double SqCir::getPerimeter() const {
25     return 4 * a + 2 * M_PI * r;
26 }
```

code of SqCirTest.cpp

```
1  #include <iostream>
2  #include <string>
3  #include "SqCir.hpp"
4  using namespace std;
5
6  //sample output:
7  //area: 18.635
8  //perimeter: 19.708
9  int main() {
10     SqCir shape(1, 2.5);
11     cout << "area: " << shape.getArea() << endl;
12     cout << "perimeter: " << shape.getPerimeter() << endl;
13
14     return 0;
15 }
```

6 (10 points) function on vectors

Define a function called `fourOrMoreSucc`, given a vector of chars `v` and a char `toAdd`, do the following:

(1) Push `toAdd` to the back of `v` using `push_back` method of vector.

(2) Test whether there were 4 or more **consecutive** elements in the **back** of the vector. If so, return true, otherwise, return false.

For example, if the vector has elements `{'r', 'b', 'r'}`, and the element to add is `'r'`, then the return is false. Reason: after pushing `'r'` to the back of the vector, the elements change to `{'r', 'b', 'r', 'r'}`, but there are only two **consecutive** `'r'` in the **back** of the vector.

If the vector has elements `{'r', 'b', 'r', 'r', 'r'}`, and the element to add is `'r'`, then the return is true. Reason: after pushing `'r'` to the back of the vector, the elements change to `{'r', 'b', 'r', 'r', 'r', 'r'}`, and there are four **consecutive** `'r'` in the **back** of the vector.

Answer: function compare is defined as follows.

```
1 bool fourOrMoreSucc(vector<char> v, char toAdd) {
2     v.push_back(toAdd);
```

```
3
4       int count = 0;
5       for (int i = v.size() - 1; i >= 0 && v[i] == toAdd; i--)
6           count++;
7
8       return (count >= 4);
9   }
```

A complete code is as follows.

```
1   #include <iostream>
2   #include <string>
3   #include <vector>
4   using namespace std;
5
6   bool fourOrMoreSucc(vector<char> v, char toAdd);
7
8   int main() {
9       vector<char> v = {'r', 'b', 'r'};
10      cout << boolalpha << fourOrMoreSucc(v, 'r') << endl; //false
11
12      vector<char> v2 = {'r', 'b', 'r', 'r', 'r'};
13      cout << boolalpha << fourOrMoreSucc(v2, 'r') << endl; //true
14
15      return 0;
16  }
17
18  bool fourOrMoreSucc(vector<char> v, char toAdd) {
19      v.push_back(toAdd);
20
21      int count = 0;
22      for (int i = v.size() - 1; i >= 0 && v[i] == toAdd; i--)
23          count++;
24
25      return (count >= 4);
26  }
```

7 (10 points) Define recursive function

Define a recursive function `reverse`, given an array of **double** with size many elements, reverse its elements. That is, swap the first and the last elements, swap the second and second to last elements, and so on. The return type is `void`.

For example, if an array with elements 1.1, 2.2, and 3.3, after the reverse, the array becomes 3.3, 2.2, 1.1

Warning: If you do not use recursion, you will not get any point.

No repetition statement, global or static variables are allowed in this function.

Use array, not vector.

Answer: Code of function is as follows.

```
void reverse(double arr[], int size) {
    if (size <= 1)
       return;

    //size >= 2
    swap(arr[0], arr[size-1]);
    reverse(arr+1, size-2);
}
```

A complete code is as follows.

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

void reverse(double arr[], int size);

int main() {
    double arr[] = {1.1, 2.2, 3.3};
    int size = sizeof(arr) / sizeof(arr[0]);

    reverse(arr, size);

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

    cout << endl;

    return 0;
}

void reverse(double arr[], int size) {
    if (size <= 1)
       return;

    //size >= 2
    swap(arr[0], arr[size-1]);
    reverse(arr+1, size-2);
}
```