1. Section, Date and Time:
   TIC-TAC, Due **05-23-2022 at 11:55 PM**

Full Name in **CAPITAL LETTERS** | **SFSU ID**

2. Fina Exam (1 exam, 0 dropped): 100 points

3. To prepare for this exam, please review all the related materials including the packages, slides, mock-up exam(s), reading assignments, in-class practices, sample programs posted in the File Manager, and assignments.
4. You do not need to print this exam. No papers. No handwriting. No scanned images. No screenshots. Please type up all your answers in the answer space available in the exam. The provided exam will be in Microsoft Word format. Please submit a single PDF via iLearn.
5. All the rules of an actual exam apply to this exam such as: closed books, closed notes, closed IDEs, and no communication with anyone except the course instructor. The course instructor will be available on Zoom (zoom.ducta.net) or via email during the exam time. You cannot use any other materials or tools but only the provided exam which will be in Microsoft Word format.
6. Please ask all your questions, if any, during the review sessions. Thank you.

HONOR CODE:

- Please follow the CS Department's policies: https://cs.sfsu.edu/student-policies
- Please follow the course's policies: http://csc340.ducta.net/00-README-StudentConduct_AcademicHonesty.pdf

---

PART A – 50 Points

A.1 - 5 pts – *Your answer must be in your own words, be in complete sentences, and provide very specific details to earn credit.*

A Bandai Namco interviewer asks you "Why Smart Pointers?" Please start your explanation with a code segment which properly demonstrates your points.

```
shared_ptr<CSC340> sPtr { make_unique<CSC340>() };
```

Should we expect an error? Why?

A.2 - 5 pts – *Your answer must be in your own words, be in complete sentences, and provide very specific details to earn credit.*

A **`weak pointer`** is monitoring an object. Please explain in detail the steps you will take to finally have the monitored object owned by a **`unique pointer`**.

How is **memory leak** different from **dangling pointer?** Please recommend the best practice to avoid both **memory leak** and **dangling pointer**.

A.3 - 5 pts – *Your answer must be in your own words, be in complete sentences, and provide very specific details to earn credit.*

It is legal to have a **pure virtual destructor** in a **C++** class as far as that destructor has a function body. Please explain the logic behind this requirement.

Pure virtual functions in C++ can have function body implementation. Please explain a scenario when and how this implementation is used. Then please provide a sample code to access/invoke that original function code if it was overridden by a child class.

A.4 - 10 pts – *Your answer must be in your own words, be in complete sentences, and provide very specific details to earn credit.*

…

```
int funcB(int);

int funcA(int n) {

    if (n <= 0)
        return 1;
    else
        return n + funcB(n - 2);
}

int funcB(int n) {

    if (n <= 3)
        return 1;
    else
        if (n > 5) {
            return n * funcA(n - 9);
        }
        else {
            return n + funcA(n - 1);
        }
}

int main() {

    for (int i = 1; i < 4; i++) {
        cout << funcA(i) << endl;
    }

    return 0;
}
```

What is the output of this program?

Please explain your work and your answer.

A.5 - 5 pts – *Your answer must be in your own words, be in complete sentences, and provide very specific details to earn credit.*

Each lambda can have **6** components. Please name the **4** optional components.

[ ] P1:

[ ] P2:

[ ] P3:

[ ] P4:

Please write a lambda which has **5** components and returns a **`function pointer`**. Please explain your work and your answer.

Compare in detail: **lambda closure** vs. **lambda class**

Do you think it is possible to write a recursive lambda expression? Please explain in detail.

A.6 - 10 pts – *Your answer must be in your own words, be in complete sentences, and provide very specific details to earn credit.*

How is **Merge Sort** different from **Quick Sort**?

71 97 79 15 63 96 36 51 62 18 17

Show the contents of the array above **each time** a **Merge Sort** changes it while sorting the array into **ascending order**. Please explain your work and your answer.

A.7 - 10 Points – *Your answer must be in your own words, be in complete sentences, and provide very specific details to earn credit.*

- Memory Area **1**: Environment
- Memory Area **2**: Runtime Stack
- Memory Area **3**: Free-store
- Memory Area **4A**: Uninitialized Data
- Memory Area **4B**: Initialized Data
- Memory Area **5**: Binary Program

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

static string model;

class Car {
public:
    Car() {};
    static string carModel;
    string name{ "McQueen" };
};

string Car::carModel = "S";

Car* testCar(string str) {
    unique_ptr<Car> uCarPtr { make_unique<Car>() };
    static Car car;
    Car* driverlessCar = new Car();
    return &car;
}

function<int (void)> testLambda() {

    int price = 100000;
    int rank = 1;

    function<int(void)> carLambda = [rank, &price]()->int {
        cout << "carLambda in testLambda" << endl;
        return price + rank;
    };

    return carLambda;
}

int main(int argc, char* argv[], char* envp[]) {
    Car* carPtr = testCar("CS");
    string carName{ carPtr->name };
    model = Car::carModel;

    auto testLambdaPtr = testLambda();
    cout << testLambdaPtr() << endl;

    return 0;
}
```

a. In **which memory area** is this element stored? Please **state** your choice and explain **why**?
b. The **lifetime**, beginning & end, of this element? **Why**?

uCarPtr object **[1] [2] [3] [4a] [4b] [5]**

Why [area]?

What lifetime and why?

carLambda expression **[1] [2] [3] [4a] [4b] [5]**

Why [area]?

What lifetime and why?

testLambdaPtr closure **[1] [2] [3] [4a] [4b] [5]**

Why [area]?

What lifetime and why?

envp [1] [2] [3] [4a] [4b] [5]

Why [area]?

What lifetime and why?

PART B – 50 Points

B.1 - 10 Points – *Your answer must be in your own words, be in complete sentences, and provide very specific details to earn credit.*

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

const int y = 1;

int main() {

   int static y = 2;

   int i = 3, j = 4, m = 5, n = 6;

   int a = [](int x, int i = 1) { return x * i; } (y, 3);

   int b = [=](int x) { return [=](int b) { return b + j; }(x) % 7; }(a);

   int c = [=](int x) mutable ->int {

      m = 6;

      return [&](int j) mutable {
         y = a * b;
         return y / j;

      }(x)-m;

   }(b);

   cout << a << endl;
   cout << b << endl;
   cout << c << endl;
   cout << m << endl;
   cout << y << endl;

   return 0;
}
```

This program **outputs 5 lines**. What are they? Please explain your work and your answer.

1.

2.

3.

4.

5.

B.2 - 10 Points – *Your answer must be in your own words, be in complete sentences, and provide very specific details to earn credit.*

```
Installer* make(const Installer& i, const double& s) {
   unique_ptr<Installer> u{ make_unique<Installer>(i, s) };
   return u.release();
}
```

Please use 5 different approaches to create **function pointer `funcPtr`** which points to **`make`**. Please explain your work and your answer.

auto

Actual type

Function object, <functional>

typedef

using

B.3 - 10 Points – *Your answer must be in your own words, be in complete sentences, and provide very specific details to earn credit.*

```
int wakeDaemon(const int& pId, Daemon* pAddress) {
   return 60;
}

string wakeDaemon(const int& pId, int tVal) {
   return "pId: " + to_string(60 * tVal);
}
```

Please create 2 function pointers:

- fPtr1 points to the first **wakeDaemon**
- fPtr2 points to the second **wakeDaemon**

Please explain your work and your answer.

First

Second

B.4 - 10 Points – *Your answer must be in your own words, be in complete sentences, and provide very specific details to earn credit.*

```
unique_ptr<PC> name_uPtr { make_unique<PC>("accountId") };
```

Please write 1 function header which can help properly pass the above unique pointer in main to a function and explain how it works in detail.

Please describe a scenario when the approach you proposed above may not be the best solution then provide a better approach. And code the function header for the better approach.

B.5 - 10 Points – *Your answer must be in your own words, be in complete sentences, and provide very specific details to earn credit.*

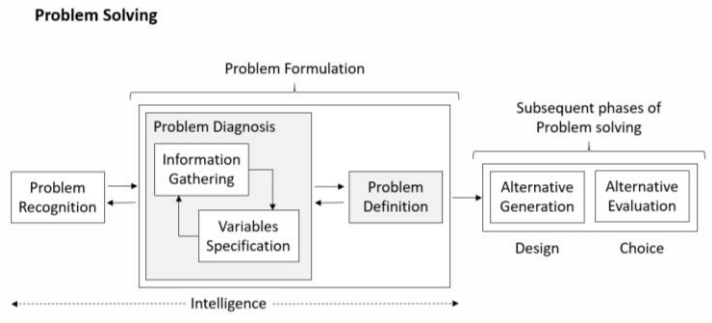
```
...
Student* func() {

    unique_ptr<Student> arr[]
    {
        make_unique<Student>("CSC340")
    };

    // #1 Insert Code
}

int main() {
    // #2 Insert Code
}
```

[**#**1 Insert Code]: **Write code** to keep all the object(s) which element(s) of array **arr** owns alive outside of the scope of **func**.

[**#**2 Insert Code]: **Write code** to have a weak_ptr monitor the object which survived; Then test if it has any owner; Then properly destroy it; Then test again if has any owner; Then destroy the Control Block.

PART C – 10 Extra Credit Points

C.1 - 5 pts



What did the IDEO team do during the Problem Formulation phase? What was the most important question? Can we use the same approach in Software Development?

C.2 - 5 pts

Your programming team lead/leader asks you to use raw pointer to implement a memory bound function. You know that using Smart Pointers is the way to go. And you heard that the leader does not know Smart Pointers. What would you do and say? Please provide your answer in detail.