midterm2

● Graded

Student

Total Points

97.5 / 104 pts

Question 1

(no title) **4** / 4 pts

✔ **– 0 pts** Correct: Guido van Rossum

Question 2

(no title) **5** / 5 pts

✔ **– 0 pts** Correct: 25

Question 3

(no title) **5** / 5 pts

✔ **– 0 pts** Correct: Compilation error at line F

Question 4

(no title) **5** / 5 pts

✔ **– 0 pts** Correct: Manager - Michael Smith

Question 5

(no title) **0** / 5 pts

✔ **– 5 pts** Parent

Question 6

(no title) **5** / 5 pts

✔ **– 0 pts** Correct: Compilation error

Question 7

(no title) **5** / 5 pts

✔ **– 0 pts** Correct: **const string* ptr; // option 3**

Question 8

(no title) **5** / 5 pts

✔ **– 0 pts** Correct: **Derived::method(Base)**

Question 9

(no title) **5** / 5 pts

✔ **– 0 pts** Correct: **Compilation error at line B**

Question 10

(no title) **5** / 5 pts

✔ **– 0 pts** Correct: **1 3 2**

Question 11

(no title) **5** / 5 pts

✔ **– 0 pts** Correct: **Parent Child Gc**

Question 12

(no title) **48.5** / 50 pts

Output operator

✔ **– 1 pt** Printing addresses for Players

✔ **– 0.5 pts** Missing / inconsistent use of **rhs**.

1

Constructor

✔ **– 0 pts** Correct

Destructor

✔ **– 0 pts** Correct

Copy Constructor

✔ **– 0 pts** Correct

Assignment operator

✔ **– 0 pts** Correct

findHighestRanked

✔ **– 0 pts** Correct

Operator bool

✔ **– 0 pts** Correct

Printed Name _____ Net ID: _____

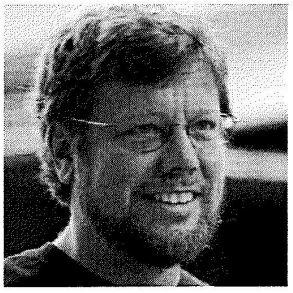
CS-UY 2124 - Object Oriented Programming
MID-TERM EXAM #2 – November 19, 2024

- Duration: 1 hour, 15 minutes
- **THE BACK OF EACH PAGE IS TO BE USED AS SCRAP PAPER, IT WILL NOT BE SCANNED INTO GRADESCOPE NOR WILL IT BE GRADED!**
- **DO NOT SEPARATE ANY PAGE. (DO NOT PULL THIS TEST APART!)**
- **PRINT YOUR FULL NAME AS IT APPEARS IN ALBERT AT THE TOP OF EVERY PAGE.**
- This is a closed-book exam. No books, notes, calculators, computers, smart watches, or phones are allowed.
- Anyone found cheating on this exam will receive a zero for the exam
- If you have a question please ask the proctor of the exam.
- Note that we have omitted any `#includes` or `using namespace std;` statements in all questions in order to save space and to save your time thinking about them. You may assume that all such statements that are needed are present. And you don't have to write them either!!!
- You also do not need to write any comments in any of your code.
- Please read all questions carefully! They may look familiar and yet be completely different.
- Answering the short-answer questions, in particular, requires that you read and understand the programs shown. You need to read them carefully if you are going to understand them.
- If a question asks you to write a class or a function and provides you with test code, **be sure your class / function works with that test code**. If the question provides you with sample output, then your answer should match that output.
- Print your name and Net ID on the top of **EACH** page. (Yes, I know we already said that.)

Printed Name ____ _____ Net ID: _ _____

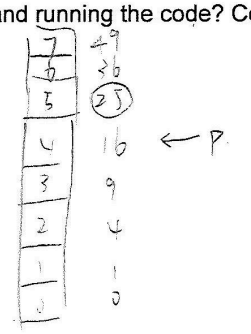
1. **EXTRA CREDIT:** Who is the creator of the Python programming language? Completely fill the circle next to your choice.

O Larry Wall
● Guido van Rossum
O Ada Lovelace
O John McCarthy
O James Gosling
O Bjarne Stroustrup
O Dennis Ritchie
O Niklaus Wirth
O Gary Kildall
O None of the above



2. **(5 pts.)** Given the code below, what is the result of compiling and running the code? Completely fill the circle next to your choice.

```
const int SIZE = 8;
int main() {
    int* arr = new int[SIZE];
    for (int i = 0; i < SIZE; i++) {
      arr[i] = i * i;
    }
    int* p = arr + SIZE - 4;
    cout <<  p[1] << endl;
}
```



O The program compiles, runs and outputs: 4
O The program compiles, runs and outputs: 9
O The program compiles, runs and outputs: 16
● The program compiles, runs and outputs: 25
O The program compiles, runs and outputs: 36
O The program compiles but has a run-time error
O Compilation error
O None of the above

Printed Name __ _____ Net ID: _____

3. **(5 pts.)** Given the code below, what is the result of compiling and running the code? Completely fill the circle next to your choice.

```
class Parent {
public:
    virtual void display() { cout << "Parent "; }          // Line A
};

class Child : public Parent {
public:
    void display() const { cout << "Child "; }             // Line B
};

class Grandchild : public Child {
public:
    void display() { cout << "Grandchild "; }              // Line C
};

int main() {
    Child c;                                               // Line D
    Grandchild gc;                                         // Line E
    const Parent* par = &gc;                               // Line F
    par->display();
}
```

does gc have const method display? no!

- O The program will output: `Child Grandchild`
- O The program will output: `Parent`
- O The program will output: `Child`
- O The program will output: `Grand Child`
- O The program will compile and not output anything
- O The program will compile, but will crash when run
- O Compilation error at Line A
- O Compilation error at Line B
- O Compilation error at Line C
- O Compilation error at Line D
- O Compilation error at Line E
- ● Compilation error at Line F
- O All of the above
- O None of the above

Printed Name ___ _____ Net ID: _____

4. **(5 pts.)** Given the code below, what is the result of compiling and running the code? Completely fill the circle next to your choice.

```
class Employee {
public:
    Employee(const string& name, double salary) : name(name), salary(salary) { }
    virtual void foo() const {
        cout << "Employee - ";
        display();
    }
    void display() const {
        cout << "name: " << name << ", salary: " << salary << endl;
    }
private:
    string name;
    double salary;
};

class Manager : public Employee {
public:
    Manager(const string& name, double salary) : Employee(name, salary) { }
    void foo() const {
        cout << "Manager - ";
        display();
    }
};

void someFunc(Employee& e) {
    e.foo();
}
int main() {
    Manager m("Michael Smith", 20000);
    someFunc(m);
}
```

O Employee - name: Michael Smith, salary: 20000

O Employee - name: Michael Smith, salary: 20000
Manager - name: Michael Smith, salary: 20000

O Manager - name: Michael Smith, salary: 20000
Employee - name: Michael Smith, salary: 20000

● Manager - name: Michael Smith, salary: 20000

O The program compiles but has a run-time error

O Compilation error

O None of the above

Printed Name ___ _____ Net ID: _____

5. **(5 pts.)** Given the code below, what is the result of compiling and running the code? Completely fill the circle next to your choice.

```
class Parent {
public:
    virtual void display() const = 0;                    // Line A
};

void Parent::display() const { cout << "Parent"; }       // Line B

class Child : public Parent {
public:
    void display() { cout << "Child "; }                 // Line C
};

void someFunc(Parent& val) {
    val.display();                                       // Line D
}

int main() {
    Child child;                                         // Line E
    someFunc(child);                                     // Line F
}
```

- ● Parent
- ○ Child
- ○ Compilation error at line A
- ○ Compilation error at line B
- ○ Compilation error at line C
- ○ Compilation error at line D
- ○ Compilation error at line E
- ○ Compilation error at line F
- ○ None of the above

Printed Name ___ _____ Net ID: _____

6. **(5 pts.)** Given the code below, what is the result of compiling and running the code? Completely fill the circle next to your choice.

```
class Parent {
public:
    Parent(int x) : x(x) {}
    virtual void display() const {
        cout << "x:" << x << endl;
    }
private:
    int x;
};

class Child : public Parent {
public:
    Child(int y) : y(y) {}
    void display() const {
        cout << "y:" << y << endl;
    }
private:
    int y;
};

void someFunc(Parent& val) {
    val.display();
}

int main() {
    Child child(10);
    someFunc(child);
}
```

and x(0)

but Parent don't have a default constructor

override

- O x:10
- O x:10 y:10
- O y:10
- ● Compilation Error
- O run-time error
- O none of the above

Printed Name ___ _____ Net ID: _ ___

7. **(5 pts.)** Given the following function definition:

```
void func(const string& name) {
    // definition for the variable ptr goes here...

    ptr = &name;
}
```

Which of the following definitions for the local variable `ptr` will allow the function `func` to successfully compile? Completely fill the circle next to your choice.

- O `string* const ptr; // option 1`
- O `string const ptr; // option 2`
- ● `const string* ptr; // option 3`
- O `const string& ptr; // option 4`
- O `string ptr*; // option 5`
- O option 1 and option 2
- O option 3 and option 4
- O options 1 - 5
- O None of the above

Printed Name _____ Net ID: ____

8. **(5 pts.)** Given the code below, what is the result of compiling and running the code? Completely fill the circle next to your choice.

```
class Derived;  // Yes, we need this.

class Base {
public:
    virtual void method(Base& arg) {  cout << "Base::method(Base)\n"; }
    virtual void method(Derived& arg) {
        cout << "Base::method(Derived)\n";
    }
};

class Derived : public Base {
public:
    void method(Base& arg) { cout << "Derived::method(Base)\n"; }
    void method(Derived& arg) { cout << "Derived::method(Derived)\n"; }
};

void someFunc(Base& argA, Base& argB) {
    argA.method(argB);
}

int main() {
    Derived d;
    Base b;
    someFunc(d, d);
}
```

- O Base::method(Base)
- O Base::method(Derived)
- ● Derived::method(Base)
- O Derived::method(Derived)
- O The program fails to compile
- O A runtime error (or undefined behavior)
- O None of the above

Printed Name ___ _____ Net ID: _____

9. **(5 pts.)** Given the following code, what is the result of compiling and running the program? Completely fill the circle next to your choice.

```
class Pet {
public:
    Pet(const string& name) : name(name) {}
protected:
    string name;
};

class Cat : public Pet {
public:
    Cat(const string& name) : Pet(name) {}
    void attack(const Cat& acat) {
        cout << "Cat " << name << " attacking Cat " << acat.name;     // Line A
    }
};

class Dog : public Pet {
public:
    Dog(const string& name) : Pet(name) {}
    void attack(const Cat& acat) {
        cout << "Dog " << name << " attacking Cat " << acat.name;     // Line B
    }
};

int main() {
    Cat felix("Felix");                                              // Line C
    Dog elmo("Elmo");                                                // Line D
    Dog apollo("Apollo");                                            // Line E
    elmo.attack(felix);                                              // Line F
}
```

- O `Dog Elmo attaching Dog Apollo`
- O `Dog Apollo attaching Dog Elmo`
- O `Dog Elmo attaching Cat Felix`
- O The program compiles and crashes when run.
- O Compilation error at Line A
- ● Compilation error at Line B
- O Compilation error at Line C
- O Compilation error at Line D
- O Compilation error at Line E
- O Compilation error at Line F
- O Compilation errors at Lines A and B
- O None of the above

Printed Name _____ Net ID: _____

10. **(5 pts.)** Given the code below, what is the result of compiling and running the code? Completely fill the circle next to your choice.

```
class Parent {
public:
    Parent(int x) : x(x) { cout << x << " "; }          // Line A
private:
    int x;
};

class Member {
public:
    Member(int m) : m(m) { cout << m << " "; }          // Line B
private:
    int m;
};

class Child : public Parent  {
public:
    Child (int x, int y, int z) : y(y), z(z), Parent(x) {}  // Line C
private:
    Member z;
    Member y;
};

int main() {
    Child c(1, 2, 3);                                  // Line D
}
```

when Child initializing.
Parent(x) → Parent(1) is initialized first. then. y and z.
cout << 1 << " "
I must be first
but z and y?

- O 3 2 1
- O 1 2 3
- ● 1 3 2
- O 1 1 1
- O 1 2 2
- O Undefined output
- O program crashes
- O Compilation error at Line A
- O Compilation error at Line B
- O Compilation error at Line C
- O Compilation error at Line D
- O None of the above

if I remember correct.
field initialized first
then constructor.

Printed Name _____ _____ Net ID: _____

11. **(5 pts.)** Given the following code, what is the result of compiling and running the program? Completely fill the circle next to your choice.

```
class Parent {
public:
    virtual void display() const { cout << "Parent "; }
};

class Child : public Parent {
public:
    void display() const { cout << "Child "; }
};

class Gc : public Child {
public:
    void display() const { cout << "Gc "; }
};

int main() {
    vector<Parent*> vp;
    vp.push_back(new Parent);
    vp.push_back(new Child);
    vp.push_back(new Gc);
    for (const Parent* ptr : vp) ptr->display();
}
```

- O Parent Parent Parent
- O Child Child Child
- O Gc Gc Gc
- O Gc Parent Child
- ● Parent Child Gc
- O Parent Parent Child
- O Parent Child Child
- O The program runs but causes a run-time error
- O The program fails to compile
- O None of the above

Printed Name __ _____ Net ID: _____

12. **(50 pts.)** The following problem involves 3 classes: **Tournament**, **MasTourn** and **Player**. You are only responsible for implementing the **MasTourn** (Masters Tournament) class.

Tournament class

- defines a constructor accepting the name of the Tournament
- supports copy control
- has a private `string` member representing the name of the Tournament object
- has a **getName()** method that returns the name of the Tournament
- may or may not have additional fields and methods, so do not assume additional methods that are not explicitly mentioned.

Note: you are **not** responsible for implementing the **Tournament** class

The Player class:

- supports **copy control** and **all necessary operators**.
  - This includes relational operators (<, <=, ==, !=, > and >=) that compare two players based on the rating of each.
- A Player has a name (`string`) and a rating (`int`) and a constructor to initialize them.
- **May** have additional fields that you don't know about.
- **Do not assume** the existence of any other methods for the `Player` class

Note: You are **not** responsible for implementing the `Player` class.

MasTourn class

- `MasTourn` is a derived class of the `Tournament` class.
- The `MasTourn` class contains the following data members:
  - a `string` representing the country where the Masters Tournament is played.
  - an `int` representing the division where the Masters Tournament is played.
  - a `vector` of **`Player`** pointers
    - All of the **`Player`** objects in the vector will be stored on the heap.
    - The **`Player`** objects are "owned by" the **MasTourn** instance, i.e. no one else has a pointer to these **`Player`** objects.
    - The **`Player`** objects in the vector represent the Tournament participants.
- A constructor that takes the MasTourn's **name**, **country** and **division**.
- Copy control. Yes, **all** of it!
- An **output operator**. You may choose the format, but the name, country, division and the players in the masters tournament should all be displayed clearly.
- a **`findHighestRanked()`** method that returns a pointer to the player with the highest rating. Returns nullptr if there are no players.
- an `addPlayer` method that takes in the player's name and rating, creating the Player object on the heap and adding it to the `MasTourn`'s vector. To save you a few lines of code, **you are not responsible for implementing this method**. You will **NOT** use this method in *your* code.

Implement the `MasTourn` class so that it satisfies the above requirements **and satisfies the test code** on the following page:

[Test code and output are on the next page]

Printed Name __ _____ Net ID: _____

```
int main() {
    MasTourn MasTournA("Grand Masters", "Austria", 1);
    MasTourn MasTournC("Masters Montreal", "Canada", 3);

    if (MasTournA) { // How does this work?
        cout << MasTournA.getName() << " has players!\n";
    } else {
        cout << MasTournA.getName() << " has no players yet.\n";
    }

    MasTournA.addPlayer("Moe", 2);
    MasTournA.addPlayer("Larry", 5);
    MasTournA.addPlayer("Curly", 3);
    cout << MasTournA << endl;

    if (MasTournA) {
        cout << MasTournA.getName() << " has players!\n";
    } else {
        cout << MasTournA.getName() << " has no players yet.\n";
    }

    cout << MasTournA.getName() << "'s top ranked player is: "
         << *MasTournA.findHighestRanked() << endl;
}
```

Sample Output:

```
Grand Masters has no players yet.
Grand Masters, Austria, 1:
Player: Moe with a rating of 2
Player: Larry with a rating of 5
Player: Curly with a rating of 3

Grand Masters has players!
Grand Masters's top ranked player is: Player: Larry with a rating of 5
```

Begin your implementation of the **MasTourn** class **on the next page**.

Printed Name ___ _____ Net ID: _____

Continue your implementation of the **MasTourn** class on this page.

```
class MasTourn : public Tournament{
private:
        string country;
        int division;
        vector<Player*> players;
public:
    MasTourn(const string& name, const string& country, int division):
    Tournament(name), country(country), division(division) {}
    virtual ~MasTourn() {
        for (size_t i=0; i<players.size(); i++){
            delete players[i];
            players[i]= nullptr;
        }
        players.clear();
    }
    MasTourn (const MasTourn & rhs): Tournament(rhs), country(rhs.country),
    division(rhs.division) {
        for (size_t i=0; i<players.size(); i++){
            players.push_back(new Player(*rhs.players[i]));
        }
    }
MasTourn& operator=(const MasTourn& rhs) {
    if (this != &rhs) {
        for (size_t i=0; i<players.size(); i++) {
            delete players[i];
            players[i]= nullptr;
        }
        players.clear();
        Tournament::operator=(rhs);
        for (size_t i=0; i<rhs.players.size(); i++){
            players.push_back(new Player(*rhs.players[i]));
        }
        country = rhs.country;
        division = rhs.division;
    }
```

Printed Name ___ _____ Net ID: _____

Continue your implementation of the **MasTourn** class on this page.

```
const Player * findHighestRanked( ) const( ) {
        if (players.size( ) == 0) { return nullptr; }
        const Player * highest = players[0];
        for (size_t i = 0; i < players.size( ); i++) {
            if (*highest <= *(players[i])) {
                highest = players[i];
            }
        }
        return highest;
}

explicit operator bool( ) const {
        return (players.size( ) != 0);
}
private:
    friend ostream& operator << (ostream& os, const MasTourn& rhs) {
        if (!rhs) { return os << rhs.getName( ) << " has no players yet."; }
        os << rhs.getName( ) << ", " << rhs.country << ", " << rhs.division << ":";
        for (size_t i = 0; i < players.size( ); i++) {
            os << endl << players[i];
        }
        return os;
    }
};
```

Printed Name ___ _____ Net ID: _ ___

Continue your implementation of the **MasTourn** class on this page.