**Lab 11**

**Object Oriented Programming Lab**

**Common Solution**

**14 Marks**

**Challenge-1: (14)**

**Creature.h**

#ifndef CREATURE\_H

#define CREATURE\_H

#include <cstdlib>

#include <ctime>

#include"String.h"

class Creature

{

protected:

int strength;

int hitpoints;

public:

Creature();

Creature(int newStrength, int newHit);

int getDamage() const;

int getStrength() const;

int getHitpoints() const;

void setStrength(const int newStrength);

void setHitpoints(const int newHit);

};

#endif

**Creature.cpp**

#include"Creature.h"

Creature::Creature()

{

strength = 10;

hitpoints = 10;

}

Creature::Creature(int newStrength, int newHit)

{

strength = newStrength;

hitpoints = newHit;

}

int Creature::getDamage() const -----(2)

{

return (rand() % strength) + 1;

}

int Creature::getStrength() const

{

return strength;

}

int Creature::getHitpoints() const

{

return hitpoints;

}

void Creature::setStrength(const int newStrength)

{

strength = newStrength;

}

void Creature::setHitpoints(const int newHit)

{

hitpoints = newHit;

}

**Human.h**

#ifndef HUMAN\_H

#define HUMAN\_H

#include"Creature.h"

class Human : public Creature

{

public:

Human();

Human(int newStrength, int newHit);

};

#endif

**Human.cpp**

#include"Human.h"

Human::Human() : Creature()

{

}

Human::Human(int newStrength, int newHit) : Creature(newStrength, newHit) -----(0.5)

{

}

**Demon.h**

#ifndef DEMON\_H

#define DEMON\_H

#include"Creature.h"

class Demon : public Creature

{

public:

Demon();

Demon(int newStrength, int newHit);

int getDamage() const;

};

#endif

**Demon.cpp**

#include"Demon.h"

Demon::Demon() : Creature()

{

}

Demon::Demon(int newStrength, int newHit) : Creature(newStrength, newHit) -----(0.5)

{

}

int Demon::getDamage() const -----(3)

{

int damage = Creature::getDamage();

if (rand() % 100 < 5)

{

damage += 50;

cout << "Demonic attack! Extra 50 damage inflicted!" << '\n';

}

return damage;

}

**Elf.h**

#ifndef ELF\_H

#define ELF\_H

#include"Creature.h"

class Elf : public Creature

{

public:

Elf();

Elf(int newStrength, int newHit);

int getDamage() const;

};

#endif

**Elf.cpp**

#include"Elf.h"

Elf::Elf() : Creature()

{

}

Elf::Elf(int newStrength, int newHit) : Creature(newStrength, newHit) -----(0.5)

{

}

int Elf::getDamage() const -----(3)

{

int damage = Creature::getDamage();

if (rand() % 100 < 10)

{

damage \*= 2;

cout << "Magical attack! Damage doubled!" << '\n';

}

return damage;

}

**Cyberdemon.h**

#ifndef CYBERDEMON\_H

#define CYBERDEMON\_H

#include"Demon.h"

class Cyberdemon : public Demon

{

public:

Cyberdemon();

Cyberdemon(int newStrength, int newHit);

};

#endif

**Cyberdemon.cpp**

#include"Cyberdemon.h"

Cyberdemon::Cyberdemon() : Demon()

{

}

Cyberdemon::Cyberdemon(int newStrength, int newHit) : Demon(newStrength, newHit) -----(1)

{

}

**Balrog.h**

#ifndef BALROG\_H

#define BALROG\_H

#include"Demon.h"

class Balrog : public Demon

{

public:

Balrog();

Balrog(int newStrength, int newHit);

int getDamage() const;

};

#endif

**Balrog.cpp**

#include"Balrog.h"

Balrog::Balrog() : Demon()

{

}

Balrog::Balrog(int newStrength, int newHit) : Demon(newStrength, newHit) -----(0.5)

{

}

int Balrog::getDamage() const -----(3)

{

int damage = Demon::getDamage();

cout << "Balrog attacks for " << damage << " points!" << '\n';

int damage2 = Demon::getDamage();

cout << "Balrog attacks for another " << damage2 << " points!" << '\n';

return damage + damage2;

}

**main**

#include<iostream>

#include <cstdlib>

#include <ctime>

#include"Human.h"

#include"Cyberdemon.h"

#include"Balrog.h"

#include"Elf.h"

using namespace std;

int main()

{

srand(time(0));

Creature creature;

Demon demon;

Elf elf;

Human human;

Cyberdemon cyberdemon;

Balrog balrog;

cout << "Creature damage: " << creature.getDamage() << '\n';

cout << "Demon damage: " << demon.getDamage() << '\n';

cout << "Elf damage: " << elf.getDamage() << '\n';

cout << "Human damage: " << human.getDamage() << '\n';

cout << "Cyberdemon damage: " << cyberdemon.getDamage() << '\n';

cout << "Balrog damage: " << balrog.getDamage() << '\n';

return 0;

}

**Quick Revision:**

**Challenge-1: (14)**

int Creature::getDamage() const -----(2)

For finding random damage correct

If damage can be zero (not added one after using rand) 🡺 -1

Human::Human(int newStrength, int newHit) -----(0.5)

If written getDamage() function in Human 🡺 -1

Demon::Demon(int newStrength, int newHit) -----(0.5)

int Demon::getDamage() const -----(3)

For using Creature function 🡺 1

For condition of 5% and adding in damage 🡺 2

Elf::Elf(int newStrength, int newHit) -----(0.5)

int Elf::getDamage() const -----(3)

For using Creature function 🡺 1

For condition of 10% and double the damage 🡺 2

Cyberdemon::Cyberdemon(int newStrength, int newHit) -----(1)

If written getDamage() function in Cyberdemon 🡺 -1

Balrog::Balrog(int newStrength, int newHit) -----(0.5)

int Balrog::getDamage() const -----(3)

For using Demon function 🡺 1

If again written 5% condition 🡺 -1

For second damage 🡺 2

**Penalty Matrix:**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Penalty List | Labs | | | | | | | | | | | | | | | |
| 1 | 2 | 3 | 3 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| Indentation putting { Infront of loop header, in do while, putting while with closing } | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  | 0 | 0 |  |  |  |  |  |
| Meaningful Variable Names |  | -2 | -2 | -2 | -2 | -2 | -2 | -2 |  | -2 | -2 |  |  |  |  |  |
| Camel Case Notation | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  | 0 | 0 |  |  |  |  |  |
| Atomicity |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Syntax error | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  | 0 | 0 |  |  |  |  |  |
| Linker error | 0 | 0 | 0 | 0 | 0 | 0 | - | - |  | - | - |  |  |  |  |  |
| Wrong function prototypes | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  | 0 | 0 |  |  |  |  |  |
| Class interface or additional members |  |  | 0 | 0 | 0 | 0 | 0 | 0 |  | 0 | 0 |  |  |  |  |  |
| Use of library function/class without permission | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  | 0 | 0 |  |  |  |  |  |
| Continue statement | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  | 0 | 0 |  |  |  |  |  |
| cin/cout where it isn’t needed | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  | 0 | 0 |  |  |  |  |  |
| Multi-filing |  |  | 0 | 0 | 0 | 0 | 0 | 0 |  | 0 | 0 |  |  |  |  |  |
| Wrong #ifndef or name of header file |  |  | -2 | -2 | -2 | -2 | -2 | -2 |  | -2 | -2 |  |  |  |  |  |
| Global functions |  |  | -3 | -3 | -3 | -3 | -3 | -3 |  | -3 | -3 |  |  |  |  |  |
| Multiple classes in one header file |  |  | -3 | -3 | -3 | -3 | -50% | -50% |  | -50% | -50% |  |  |  |  |  |