**Lab Summary:**

This lab was all about a review of classes. Also, we used pointers to classes and implementing arrays of classes and pointers to classes. This lab made us take a good look at memory location and how it is stored within the system. This is done by looking at the hexadecimal address that is provided with the sizeof operator.

**Header File:**

#ifndef PAYROLL\_H\_

#define PAYROLL\_H\_

#include <string>

class PayRoll {

private:

std::string name;

int payrate;

double hours;

public:

PayRoll();

PayRoll(std::string, int);

void setHours(double);

double calculatePay();

void printInfo();

std::string getName();

};

#endif

**Implementation File:**

#include "payroll.h" // Header file

#include <iostream> // cout, endl

#include <string>

/\* Ryan Rosiak

\*

\* Implementaion file for payroll.h:

\*

\*/

/\*

\* Default Constructor:

\* Initializes all members to default values.

\*/

PayRoll::PayRoll() {

name = "";

payrate = 0;

hours = 0.0;

}

/\*

\* Constructor:

\* Takes a string and int as arguments and assigns them to

\* name and payrate.

\*/

PayRoll::PayRoll(std::string n, int pr) {

name = n;

payrate = pr;

}

/\*

\* Setter:

\* Takes an double as an arugment and assigns it to hours

\*/

void PayRoll::setHours(double h) {

hours = h;

}

/\*

\* Calculate Function:

\* Calculates the total pay by returning the value of hours

\* multiplied by pay

\*/

double PayRoll::calculatePay() {

return hours \* payrate;

}

/\*

\* PrintInfo Function:

\* Displays all possible info in given class

\*/

void PayRoll::printInfo() {

std::cout << "Name: " << name << std::endl;

std::cout << "PayRate: " << payrate << std::endl;

std::cout << "Hours: " << hours << std::endl;

std::cout << "Total Pay: " << calculatePay() << std::endl;

}

/\*

\* Getter:

\* Returns a string that is the name of said object

\*/

std::string PayRoll::getName() {

return name;

}

**Main:**

#include "payroll.h" // Class file

#include <iostream> // cout, cin, endl

int main() {

int inHour; // Input variable for hours worked

bool done = false; // Flag variable for loop

PayRoll\* parray[7];

PayRoll parray2[7];

PayRoll p1 {"Bob", 12};

PayRoll p2 {"Ryan", 15};

PayRoll p3 {"Sky", 22};

PayRoll p4 {"Mary", 25};

PayRoll p5 {"Jeff", 9};

PayRoll p6 {"Arnold", 11};

PayRoll p7 {"Naomi", 18}; // Initialize all the objects

parray[0] = &p1;

parray[1] = &p2;

parray[2] = &p3;

parray[3] = &p4;

parray[4] = &p5;

parray[5] = &p6; // Assign all of the object's addresses to the pointer array

parray[6] = &p7;

PayRoll p8 {"Bob", 12};

PayRoll p9 {"Ryan", 15};

PayRoll p10 {"Sky", 22};

PayRoll p11 {"Mary", 25};

PayRoll p12 {"Jeff", 9};

PayRoll p13 {"Arnold", 11};

PayRoll p14 {"Naomi", 18};

parray2[0] = p8;

parray2[1] = p9;

parray2[2] = p10;

parray2[3] = p11;

parray2[4] = p12;

parray2[5] = p13;

parray2[6] = p14;

for (int i = 0; i < 7; i++) {

std::cout << "Enter the amount of hours worked for " << parray[i]->getName() << ": ";

std::cin >> inHour;

while (!done) {

if (inHour <= 60) { // Validates that the hours are not over 60

std::cout << "Success!" << std::endl;

parray[i]->setHours(inHour);

done = true; // Continues to next position in the array

} else {

std::cout << "Unsuccessful! Hours cannot be over 60. " << std::endl;

std::cout << "Enter the amount of hours worked for " << parray[i]->getName()

<< ": ";

std::cin >> inHour; // Gives another chance to input a correct number

std::cout << std::endl;

}

}

done = false; // Allows for repeated validation

}

for (int i = 0; i < 7; i++) { // Prints the info of all of the PayRoll objects

parray[i]->printInfo();

std::cout << std::endl;

}

std::cout << "Size of pointer array: " << std::endl;

std::cout << sizeof(parray) << std::endl;

std::cout << "Size of regular array of objects: " << std::endl;

std::cout << sizeof(parray2) << std::endl;

std::cout << "Memory addresses for pointer array: " << std::endl;

for (int i = 0; i < 7; i++) {

std::cout << parray[i] << std::endl;

}

std::cout << "Memory addresses for regular array of objects: " << std::endl;

for (int i = 0; i < 7; i++) {

std::cout << &parray2[i] << std::endl;

}

return 0;

}

/\* 9.

\* The difference between the pointer array and the regular array of objects

\* is that the pointer array is holding only the memory address that is pointing to

\* the location of the objects that are deep within it. The actual array of class objects

\* must store all of the data and bytes from each object into one array. Therefore, it is much

\* larger of an array because the object and its data members has much more memory used

\* then does just an array of a pointer pointing to memory addresses.

\*/

/\* 10.

\* The addresses are right next to each other within their own given array because that is the way

\* that an array handles incrementing subscripts and keeping objects together. Each object in memory is

\* its specificed amount of memory away from each other. This is the same for both arrays. But they both also

\* seem to be somewhat together in memory as well. In other words, both arrays are close to each other in

\* memory. That could be a coincidence but it is no coincidence that each object address in each given array

\* follows one another in memory.

\*/

**Example Output:**

Enter the amount of hours worked for Bob: 12

Success!

Enter the amount of hours worked for Ryan: 54

Success!

Enter the amount of hours worked for Sky: 69

Unsuccessful! Hours cannot be over 60.

Enter the amount of hours worked for Sky: 89

Unsuccessful! Hours cannot be over 60.

Enter the amount of hours worked for Sky: 50

Success!

Enter the amount of hours worked for Mary: 22

Success!

Enter the amount of hours worked for Jeff: 65

Unsuccessful! Hours cannot be over 60.

Enter the amount of hours worked for Jeff: 1000

Unsuccessful! Hours cannot be over 60.

Enter the amount of hours worked for Jeff: 12

Success!

Enter the amount of hours worked for Arnold: 25

Success!

Enter the amount of hours worked for Naomi: 28

Success!

Name: Bob

PayRate: 12

Hours: 12

Total Pay: 144

Name: Ryan

PayRate: 15

Hours: 54

Total Pay: 810

Name: Sky

PayRate: 22

Hours: 50

Total Pay: 1100

Name: Mary

PayRate: 25

Hours: 22

Total Pay: 550

Name: Jeff

PayRate: 9

Hours: 12

Total Pay: 108

Name: Arnold

PayRate: 11

Hours: 25

Total Pay: 275

Name: Naomi

PayRate: 18

Hours: 28

Total Pay: 504

Size of pointer array:

28

Size of regular array of objects:

280

Memory addresses for pointer array:

0x62fbd0

0x62fba8

0x62fb80

0x62fb58

0x62fb30

0x62fb08

0x62fae0

Memory addresses for regular array of objects:

0x62fbf8

0x62fc20

0x62fc48

0x62fc70

0x62fc98

0x62fcc0

0x62fce8