SCHOOL OF COMPUTER SCIENCES
UNIVERSITI SAINS MALAYSIA
Semester II, Session 2023/2024

CPT113 – Programming Methodology & Data Structure
Tutorial Week 7
Pointers and Dynamic Variables

Learning Outcomes:

- Understanding the use of pointers and dynamic variables in class

---

1. Refer to the given source code, compile and run the program. Follow the instructions in the comments and observe the effect of modifying the code on the output of the program.

```
/* This program demonstrates deep and shallow copy for objects
Instructions:
      1. Remove comment for CopyConstructor, then compile and run the program. Observe
the output. Relate the code with the output. Discuss your findings.
      2. Comment out CopyConstructor, then compile and run the program. Observe the
output.
      3. Relate with the code and output. Discuss it.

*/

#include <iostream>
using namespace std;

class ShalloC
{
private:
      int * x;
public:
      ShalloC(int m)
      {
            x = new int;
            *x = m;
      }

      //CopyConstructor
      /*ShalloC(const ShalloC& obj)
      {
            x = new int;
            *x = obj.GetX();
      } */

      int GetX() const
      {
            return *x;
```

```
	}

	void SetX(int m)
	{
		*x = m;
	}

	void PrintX()
	{
		cout << "Int X=" << *x << endl;
	}

	~ShalloC()
	{
		delete x;
	}
};

int main()
{
	ShalloC ob1(10);
	ShalloC ob2 = ob1 ;
	ob1.PrintX();
	ob2.PrintX();
	ob1.SetX(12);
	ob1.PrintX();
	ob2.PrintX();
}
```

2. Refer to the given source code, compile & run a program. Follow the instructions in the comments and observe the effect of modifying the code on the output of the program. Explain what has happened.

```
/* This program demonstrates how assignment operators work with/without dynamic array
Instructions:
1. Uncomment Method 1, then compile and run the program. Observe the output and relate
it with the code. Discuss it.
2. Comment out Method 1. Then uncomment Method 2.
3. Compile and run the program. Observe the output and relate it with the code. Discuss
it.
*/

#include <iostream>
using namespace std;

class cwork
{
	private:
		int matric;
		int *p;
	public:
		void createDynamicArray(int size)
		{
			p=new int [size];
		}
```

```
            void setData(int m,int t1,int t2)
            {
                  matric=m;
                  p[0]=t1;
                  p[1]=t2;
            }

            cwork()
            {
                  matric=0;
                  p=NULL;
            }

            ~cwork() { }

//Method 1
/* overloading of assignment operator
            void operator = (const cwork &cw){
                  matric = cw.matric;
                  p=cw.p;
            } */

//Method 2
            /* const cwork operator=(const cwork& rightObject)
            {
                  if(this != &rightObject) //avoid self-assignment
                  {
                        delete [ ] p;
                        matric = rightObject.matric;
                        p=new int [2];
                        for (int i=0;i<2;i++)
                              p[i]=rightObject.p[i];
                  }

                  //return the object assigned
                  return *this;
            } */

            void display()
            {
                  cout <<"\t" << matric <<": " ; cout <<"\t" << "Test 1 & Test 2:
            " << p[0] << "\t" << p[1] << endl;
            }

            void del()
            {
                  delete [] p;
            }
};

int main()
{
      cwork student1,student2;
      student1.createDynamicArray(2);
      student1.setData(27913, 75,80);
      cout <<"Display data in object student1: \n";
      student1.display();
```

```
        //copy data from object student1 into object student2
        student2=student1; student1.del(); //object student1 delete dynamic array
        cout <<"Display data in object student2: \n";
        student2.display();
}
```

3. Construct a class named `Health` which consists of two data members: `passportNum` with type `int` and a pointer named `ptr` with type `float`. Create a dynamic array using pointer `ptr` to store body temperature and age. You should define appropriate methods (including copy constructors, overloading of assignment operator, destructors to properly handle dynamic arrays) to complete the class definition. Then write main() function to test your program. **(let student try on their own and submit their answers)

True/False
Indicate whether the statement is true or false.

____ 4. A pointer variable is a variable whose content is a memory address.

____ 5. In C++, no name is associated with the pointer data type.

____ 6. In the statement

```
int* p, q;
```

`p` and `q` are pointer variables.

____ 7. The *address operator* is a unary operator that returns the address of its operand.

____ 8. In C++, the asterisk character is only used as the binary multiplication operator.

____ 9. In C++, `&p`, `p`, and `*p` all have the same meaning.

____ 10. In C++, the dot operator has a lower precedence than the dereferencing operator.

____ 11. Variables that are created during program execution are called `static` variables.

____ 12. In C++, `new` is a reserved word; however, `delete` is not a reserved word.

____ 13. A memory leak is an unused memory space that cannot be allocated.

____ 14. The statement `delete p;` deallocates the dynamic variable pointed to by `p`.

____ 15. Two pointer variables of the same type can be compared for equality.

____ 16. If `p` is a pointer variable, the statement `p = p * 2;` is valid in C++.

____ 17. If `p` is a pointer variable, the statement `p = p + 1;` is valid in C++.

____ 18. Pointer arithmetic is the same as ordinary arithmetic.

____ 19. Given the declarations

```
int list[10];
int *p;
```

the statement

```
p = list;
```

is valid in C++.

Multiple Choice
Identify the choice that best completes the statement or answers the question.

____ 20. In C++, you declare a pointer variable by using the ____ symbol.
a. `*`
b. `&`
c. `#`
d. `@`

____ 21. The code `int *p;` declares `p` to be a(n) ____ variable.
a. new
b. num
c. pointer
d. address

____ 22. In C++, ____ is called the address of operator.
a. `&`
b. `*`
c. `#`
d. `->`

____ 23. Which of the following correctly declares a pointer variable `p`?
a. `int *p;`
b. `int* q, p;`
c. `int p; q*;`
d. `*int p;`

____ 24. What is the value of `x` after the following statements execute?

```
int x = 25;
int *p;
p = &x;
*p = 46;
```

a. `NULL`
b. `0`
c. `25`
d. `46`

____ 25. What is the output of the following statements?

```
int x = 33;
int *q;
q = &x;
cout << *q << endl;
```

a. `NULL`
b. `0`
c. `3`
d. `33`

____ 26. What is the output of the following code?

```
int *p;
int x;
x = 76;
p = &x;
*p = 43;
cout << x << ", " << *p << endl;
```

a. `76, 76`
b. `76, 43`
c. `43, 76`
d. `43, 43`

____ 27. What is the output of the following code?

```
int *p;
int x;
x = 12;
p = &x;
cout << x << ", ";
*p = 81;
cout << *p << endl;
```

a. `12, 12`
b. `12, 81`
c. `81, 12`
d. `81, 81`

____ 28. The only number that can be directly assigned to a pointer variable is ____.
a. `-1`
b. `0`
c. `1`
d. `2`

____ 29. Which of the following can be used to initialize a pointer variable?
a. `1`
b. `NULL`
c. `"0"`
d. `'0'`

____ 30. The C++ operator ____ is used to create dynamic variables.
a. `dynamic`
b. `new`
c. `virtual`
d. `dereferencing`

____ 31. The C++ operator ____ is used to destroy dynamic variables.
a. `destroy`
b. `delete`
c. `*`
d. `~`

____ 32. Which of the following operations is allowed on pointer variables?
a. `exp`
b. `%`
c. `==`
d. `/`

____ 33. Which of the following arithmetic operations is allowed on pointer variables?
a. Increment
b. Modulus
c. Multiplication
d. Division

____ 34. Given the statement `double *p;`, the statement `p++;` will increment the value of `p` by ____ bytes.
a. one
b. two
c. four
d. eight

____ 35. Given the declaration `int *a;`, the statement `a = new int[50];` dynamically allocates an array of 50 components of the type ____.
a. `int`
b. `int*`
c. `pointer`
d. `address`

____ 36. An array created during the execution of a program is called a(n) ____ array.
a. list
b. static
c. execution
d. dynamic

____ 37. In a ____ copy, two or more pointers of the same type point to the same memory.
a. static
b. shallow
c. dynamic
d. deep

____ 38. In a(n) ____ copy, two or more pointers have their own data.
a. shallow
b. deep
c. static
d. dynamic

____ 39. A class ____ automatically executes whenever a class object goes out of scope.
a. constructor
b. destructor
c. pointer
d. exception

____ 40. The default member-wise initialization is due to the constructor, called the ____ constructor.
a. init
b. new
c. copy
d. pointer

____ 41. The ____ constructor is called when an object is passed as a (value) parameter to a function.
a. default
b. copy
c. struct
d. class

____ 42. In C++, virtual functions are declared using the reserved word ____.
a. `virtual`
b. `private`
c. `public`
d. `struct`

____ 43. In ____ binding, the necessary code to call a specific function is generated by the compiler.
a. static
b. dynamic
c. shallow
d. deep

____ 44. Run-time binding is also known as ____ binding.
a. static
b. shallow
c. dynamic
d. deep