CEC 101: Computer Programming

Civil Engineering Autumn 2023-24

Practical 7: Functions Overloading, Call by Reference, Pointers

Q1. Write a C++ program to create three overloaded functions named **Calculate().** The first function should take an integer value as a parameter representing the side of a square and return the area of the square as a double. The second function should take two decimal values representing the length and breadth of a rectangle as parameters and return the area of the rectangle as a double. The third function should take a decimal value representing the radius of a circle as a parameter and return the area of the circle as a double. Write a main function and call the above functions.

Q2. Which of the following overloaded functions are NOT allowed in C++?

1) Function declarations that differ only in the return type

```
int fun(int x, int y);
void fun(int x, int y);
```

2) Parameter declarations that differ only in a pointer * versus an array []

```
int fun(int *ptr, int n);
int fun(int ptr[], int n);
```

3) Two parameter declarations that differ only in their default arguments

```
int fun( int x, int y);
int fun( int x, int y = 10);
```

a) All of the above
b) All except 1
c) All except 2
d) All except 3

Q3. Create a function quadEquation() that calculates the solutions to quadratic equations. The function should display false, if no real solution is available, otherwise true. All argument passing through reference or pointers only.
Arguments: The coefficients a, b, c, two pointers/reference for solutions x1 and x2.

Q4. Write a function max that returns the maximum value among 3 integer numbers. The numbers should be passed by reference.

Q5. Write a program to create a dynamic array (dynamic variable concept) so that the pointer variable entry is pointing to this dynamic array which will contain 10 integers.

```
int *entry;
entry = new int[10];
```

The elements of this pointer will be initialized and printed through two separate functions which will be called from main.

Q6. Predict the output of following C++ programs/codes.

```
#include <iostream>
using namespace std;
void square (int *x) {
        *x = (*x)++ * (*x);
}
void square (int *x, int *y) {
        *x = (*x) * --(*y);
}
int main ( ){
        int number = 30;
        square(&number, &number);
        cout << number;   870
        return 0;
}
```

```
# include <iostream>
using namespace std;
void modifyValue(int* ptr2) {
        cout << ptr2 << endl;      address of num
        *ptr2 = 50;
}
int main() {
        int num = 10;
        int* ptr1 = &num;
        cout << ptr1 << endl;      address of num
        modifyValue(ptr1);
        cout <<  num << endl;      50
        return 0;
}
```

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

int fun(int=0, int = 0);                error

int main(){
        cout << fun(5);
        return 0;
}
int fun(int x, int y){
        return (x+y);
}
```

```
#include <iostream>
using namespace std;
int main() {
        int num1 = 42;
        int* ptr1 = &num1;
        cout <<  num1 << endl;          42
        cout << *ptr1 << endl;          42
        cout << ptr1 << endl;           add of num1
        *ptr1 = 100;
        cout << num1 << endl;      100
        return 0;
 }
```

```
#include <iostream>
using namespace std;
int main() {
int *p1, *p2;
p1 = new int;
p2 = new int;
*p1 = 10; *p2 = 20;
cout << *p1 << " " << *p2 << endl;      10 20
p1 = p2;
cout << *p1 << " " << *p2 << endl;      20 20
*p1 = 30;
cout << *p1 << " " << *p2 << endl;      30 30
*p1 = *p2;
cout << *p1 << " " << *p2 << endl;      30 30
*p2 = 30;
cout << *p1 << " " << *p2 << endl;      30 30
return 0;
 }
```

```
#include <iostream>
using namespace std;
int main(){
   int arr [3] = {5, 10, 20};
   int *ptr = arr;
   cout << "Elements of the array: " <<
endl;
   cout << ptr [0]<< " " << ptr [1] << " " <<
ptr[2] << endl;
   cout << ptr << endl;                 5 10 20
   cout << ptr+1 << endl;               0x61fee8
   cout << ptr+2 << endl;               0x61feec
   int x= *ptr++, y = *++ptr ;          0x61fef0
   cout << ptr << endl;                 0x61fef0
   cout <<x << " " << y << endl;        5 20
   return 0;
}
```