1.

Multiply.h

```
#ifndef MULTIPLY_H
#define MULTIPLY_H

#include <cmath>

int multiply(int a, int b) {
  return a * b;
}

double multiply(double a, double b) {
  return a * b;
}

int multiply(int a, double b, bool flag) {
  double result = a * b;
  return flag ? ceil(result) : floor(result);
}

#endif
```

Multiply.cpp

```
#include <iostream>
#include "multiply.h"

using namespace std;

int main() {
  int int1, int2;
  double double1, double2;
  bool flag;

  // Case A: Two integers
```

```
    cout << "Enter two integers: ";
    cin >> int1 >> int2;
    cout << "Product (integers): " << multiply(int1, int2) << endl;

    // Case B: Two doubles
    cout << "Enter two doubles: ";
    cin >> double1 >> double2;
    cout << "Product (doubles): " << multiply(double1, double2) << endl;

    // Case C: One integer, one double, and a flag
    cout << "Enter integer, double, and flag (1 for ceil, 0 for floor): ";
    cin >> int1 >> double1 >> flag;
    cout << "Product with flag: " << multiply(int1, double1, flag) << endl;

    return 0;
}
```

Output

```
C:\Users\nakht\OneDrive - Metropolia Ammattikorkeakoulu Oy\Documents\002\c++\hw2>g++ multiply.cpp -o multiply

C:\Users\nakht\OneDrive - Metropolia Ammattikorkeakoulu Oy\Documents\002\c++\hw2>multiply
Enter two integers: 34
45
Product (integers): 1530
Enter two doubles: 34.56
56.78
Product (doubles): 1962.32
Enter integer, double, and flag (1 for ceil, 0 for floor): 23
34.78
1
Product with flag: 800
```

Ex. 2 Ans: Swapping without pointers or references doesn't change the original values.

Using pointers or references does change the original values.

For this task, references are best—they are simple and safe to use in C++.

Code for (ex2-8)

Lab2.cpp

```
#include <iostream>
```

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

// Exercise 2: Swapping methods
void swapNoPointersOrRefs(float a, float b) {
   float temp = a; a = b; b = temp;
   cout << "After swap (no pointers/refs): a=" << a << ", b=" << b << endl;
}
void swapUsingPointers(float *a, float *b) {
   float temp = *a; *a = *b; *b = temp;
}
void swapUsingReferences(float &a, float &b) {
   float temp = a; a = b; b = temp;
}

// Exercise 3: Simple calculator
void calculator() {
   double num1, num2; char op;
   cout << "Enter first number: ";
   cin >> num1;
   cout << "Enter operator (+ - * /): ";
   cin >> op;
   cout << "Enter second number: ";
   cin >> num2;

   switch(op) {
      case '+': cout << "Result: " << num1 + num2 << endl; break;
      case '-': cout << "Result: " << num1 - num2 << endl; break;
```

```
      case '*': cout << "Result: " << num1 * num2 << endl; break;
      case '/':
        if(num2 != 0) cout << "Result: " << num1 / num2 << endl;
        else cout << "Error: Division by zero!" << endl;
        break;
      default: cout << "Invalid operator" << endl;
    }
}

// Exercise 4: Continuous numbers
void continuousNumbers() {
    int num;
    while(true) {
      cout << "Enter a number (negative to stop, zero to skip): "; cin >> num;
      if(num < 0) break;
      if(num == 0) continue;
      cout << "Square: " << num*num << endl;
    }
}

// Exercise 5: Array reverse
void reverseArray() {
    int arr[] = {1,4,7,10,15}, reversed[5], size=5;
    int *ptr = arr + size - 1;
    for(int i = 0; i < size; i++) reversed[i] = *ptr--;
    cout << "Reversed array: ";
    for(int n : reversed) cout << n << " ";
    cout << endl;
```

```
}

// Exercise 6: Student records
struct Student { char name[50]; int id; float grade; } students[100];
int count = 0;
void manageStudents() {
  int choice;
  do {
    cout << "1.Add 2.Display 3.Search 4.Exit: "; cin >> choice;
    if(choice==1) {
      cout << "Name: "; cin >> students[count].name;
      cout << "ID: "; cin >> students[count].id;
      cout << "Grade: "; cin >> students[count].grade; count++;
    } else if(choice==2) {
      for(int i=0; i<count; i++)
        cout << students[i].name << " " << students[i].id << " " << students[i].grade <<
endl;
    } else if(choice==3) {
      int sid; cout << "Search ID: "; cin >> sid;
      bool found = false;
      for(int i=0; i<count; i++)
        if(students[i].id==sid) {
          cout << students[i].name << " " << students[i].grade << endl;
          found=true;
        }
      if(!found) cout << "Not found!" << endl;
    }
  } while(choice!=4);
```

```
}

// Exercise 7: Pointers shallow copy
void shallowCopyDemo() {
   int x=100; int *p1=&x, *p2=p1;
   *p1 = 200; cout << "Pointer 2 sees: " << *p2 << endl;
}

// Exercise 8: References
void referenceDemo() {
   int a=100; int &refA=a; refA=30;
   cout << "Value of a: " << a << endl;
}

int main() {
   int task;
   do {
      cout << "\nWhich exercise (2-8) to execute? (0 to exit): ";
      cin >> task;
      switch(task) {
         case 2: {
            float x, y;
            cout << "Enter two floating values: "; cin >> x >> y;
            swapNoPointersOrRefs(x,y);
            swapUsingPointers(&x,&y); cout<<"Pointers swap: "<<x<<" "<<y<<endl;
            swapUsingReferences(x,y); cout<<"References swap: "<<x<<" "<<y<<endl;
break;
         }
```

```
            case 3: calculator(); break;
            case 4: continuousNumbers(); break;
            case 5: reverseArray(); break;
            case 6: manageStudents(); break;
            case 7: shallowCopyDemo(); break;
            case 8: referenceDemo(); break;
            case 0: cout << "Exiting..." << endl; break;
            default: cout << "Invalid choice!" << endl;
        }
    } while(task != 0);

    return 0;
}
```

Output

```
Which exercise (2-8) to execute? (0 to exit): 2
Enter two floating values: 23.4566
34.678
After swap (no pointers/refs): a=34.678, b=23.4566
Pointers swap: 34.678 23.4566
References swap: 23.4566 34.678

Which exercise (2-8) to execute? (0 to exit): 3
Enter first number: 23
Enter operator (+ - * /): +
Enter second number: 2
Result: 25

Which exercise (2-8) to execute? (0 to exit): 3
Enter first number: 45
Enter operator (+ - * /): -
Enter second number: 34
Result: 11

Which exercise (2-8) to execute? (0 to exit): 3
Enter first number: 65
Enter operator (+ - * /): *
Enter second number: 3
Result: 195

Which exercise (2-8) to execute? (0 to exit): 3
Enter first number: 345678
Enter operator (+ - * /): /2
Enter second number: Result: 172839
```

```
Which exercise (2-8) to execute? (0 to exit): 4
Enter a number (negative to stop, zero to skip): 2
Square: 4
Enter a number (negative to stop, zero to skip): 6
Square: 36
Enter a number (negative to stop, zero to skip): 0
Enter a number (negative to stop, zero to skip): -2

Which exercise (2-8) to execute? (0 to exit): 5
Reversed array: 15 10 7 4 1

Which exercise (2-8) to execute? (0 to exit): 6
1.Add 2.Display 3.Search 4.Exit: 1
Name: jack
ID: 1
Grade: 56
1.Add 2.Display 3.Search 4.Exit: 1
Name: john
ID: 2
Grade: 99
1.Add 2.Display 3.Search 4.Exit: 2
jack 1 56
john 2 99
1.Add 2.Display 3.Search 4.Exit: 3
Search ID: 2
john 99
1.Add 2.Display 3.Search 4.Exit: 4
```

```
Which exercise (2-8) to execute? (0 to exit): 7
Pointer 2 sees: 200

Which exercise (2-8) to execute? (0 to exit): 8
Value of a: 30

Which exercise (2-8) to execute? (0 to exit): 0
```