// This program demonstrates the use of the indirection operator.

/\*#include <iostream>

using namespace std;

int main()

{

int x = 25; // in variable

int \*ptr = nullptr; // Pointer variable, can point to an int

ptr = &x; // Store the address of x in ptr

// Use both x and ptr to display the value in x.

cout << "Here is the value in x, printed twice:\n";

cout << x << endl; // Displays the contents of x

cout << \*ptr << endl; // Displays the contents of x

// Assign 100 to the location pointed to by ptr. This will actually assign

// 100 to x.

\*ptr = 100;

// Use both x and ptr to display the value in x.

cout << "Once again, here is the value in x:\n";

cout << x << endl; // Displays the contents of x

cout << \*ptr << endl; // Displays the contents of x

int vals[] = { 4, 7, 11 };

cout << vals << endl; // displays 0x4a00

cout << vals[0] << endl; // displays 4

cout << \*vals << endl; // displays 4

return 0;

}

// This program shows an array name being dereferenced with the \* operator.

#include <iostream>

using namespace std;

int main()

{

short numbers[] = { 10, 20, 30, 40, 50 };

cout << "The first element of the array is ";

cout << \*numbers << endl;

return 0;

}

#include <iostream>

using namespace std;

int main()

{

const int NUM\_COINS = 5;

double coins[NUM\_COINS] = { 0.05, 0.1, 0.25, 0.5, 1.0 };

double \*doublePtr; // Pointer to a double

int count; // Array index

// Assign the address of the coins array to doublePtr.

doublePtr = coins;

// Display the contents of the coins array. Use subscripts with the pointer!

cout << "Here are the values in the coins array:\n";

for (count = 0; count < NUM\_COINS; count++)

cout << doublePtr[count] << " ";

// Display the contents of the array again, but this time use pointer

// notation with the array name!

cout << "\nAnd here they are again:\n";

for (count = 0; count < NUM\_COINS; count++)

cout << \*(coins + count) << " ";

cout << endl;

return 0;

}

#include <iostream>

using namespace std;

int main()

{

void swap(int \*x, int \*y);

int num1, num2;

cout << "Please enter your first value: ";

cin >> num1;

cout << "Please enter your second value: ";

cin >> num2;

cout << endl;

cout << "Here is the value for the first: " << num1 << endl;

cout << "Here is the value for the second: " << num2 << endl;

swap(&num1, &num2);

cout << "Here is the value for the first: " << num1 << endl;

cout << "Here is the value for the second: " << num2 << endl;

return 0;

}

void swap(int \*x, int \*y)

{

int temp;

temp = \*x;

\*x = \*y;

\*y = temp;

}

// This program uses two functions that accept addresses of variables as

// arguments.

#include <iostream>

using namespace std;

// Function prototypes

void getNumber(int \*);

void doubleValue(int \*);

int main()

{

int number;

// Call getNumber and pass the address of number

getNumber(&number);

// Call doubleValue and pass the address of number

doubleValue(&number);

// Display the value in number

cout << "That value doubled is " << number << endl;

return 0;

}

// Definition of getNumber. The parameter, input, is a pointer. This function

// asks the user for a number. The value entered is stored in the variable

// pointed to by input.

void getNumber(int \*input)

{

cout << "Enter an integer number : ";

cin >> \*input;

}

// Definition of doubleValue. The parameter, val, is a pointer. This function

// multiplies the variable pointed to by val by two.

void doubleValue(int \*val)

{

\*val \*= 2;

}

#include <iostream>

using namespace std;

void displayPayRates(const double \*, int);

int main()

{

const int size = 6;

const double payRates[size] = { 18.55, 17.45, 12.85, 14.97, 10.35, 18.89 };

const double \*rates = payRates;

displayPayRates(rates, size);

return 0;

}

void displayPayRates(const double \*rates, int size)

{

for (int count = 0; count < size; count++)

{

cout << "Pay rate for employee " << (count + 1)

<< " is $" << \*(rates + count) << endl;

}

}\*/

// This program totals and averages the sales figures for any number of days.

// The figures are stored in a dynamically allocated array.

#include <iostream>

#include <iomanip>

using namespace std;

int main()

{

double \*sales = nullptr, // to dynamically allocate an array

total = 0.0, // accumulator

average; // to hold average sales

int numDays, // to hold the number of days of sales

count; // counter variable

// Get the number of days of sales

cout << "How many days of sales figures do you wish ";

cout << "to process? ";

cin >> numDays;

// Dynamically allocate an array large enough to hold that many days of sales

// amounts.

sales = new double[numDays];

// Get the sales figures for each day.

cout << "Enter the sales figures below.\n";

for (count = 0; count < numDays; count++)

{

cout << "Day " << (count + 1) << ": ";

cin >> sales[count];

}

// Calculate the total sales

for (count = 0; count < numDays; count++)

{

total += sales[count];

}

// Calculate the average sales per day

average = total / numDays;

// Display the results

cout << fixed << showpoint << setprecision(2);

cout << "\n\nTotal Sales: $" << total << endl;

cout << "Average Sales: $" << average << endl;

// Free dynamically allocated memory

delete[] sales;

sales = nullptr; // Make sales a null pointer

return 0;

}