­­­­Given the following array definition:­­

int values[5] = {4, 7, 6, 8, 2};

What does the following statement display?

Cout << values [4] << “ “ << (values[2] + values[3])

<< “ “ << ++values [1] << end;

Answer: Nothing?? Error Unless.. 2 14 8 (gives you 7 from the index of 1 then plus 1)

1. Given the following array definition:

int numbers[5] = {1, 2, 3};

* 1. What value is stored in numbers [2]? Answer: 3
  2. What value is stored in numbers [4]? Answer: 0

1. Assume that array1 and array2 are both 25 element integer arrays. Indicate whether each of the following statements is legal or illegal and explain why:
   1. array1 = array2;   
      Answer: Cannot assign array to array
   2. cout << array1;   
      Answer: Legal, prints memory address
   3. cin >> array2;   
      Answer: Illegal, need to attach a subscript which area of array to store it in.
2. Look at the following array definition:  
   double sales [8][10];
   1. How many rows does the array have?   
      Answer: 8 Rows
   2. How many columns does the array have?   
      Answer: 10 Columns
   3. How many elements does the array have?   
      Answer: 80 elements 8 \* 10 gives total num of elements
   4. Write a statement that stores 3.52 in the last column of the last row in the array.

Answer: sales[7][9] = 3.52

1. Write value returning function that receives the above sales array and using nested loops, returns the sum of all the elements in the array.

double totalArr(double sales[][10]){

double total = 0;

for(int i = 0; i < 8; i++){

for(int j = 0; j < 10; j++){

total += sales[i][j];

}

}

*return* total;

}

1. Write a declaration for an array called names to hold 20 string elements. Using a loop, initialize all the elements to the name “Sirius Black”.

*const* int SIZE = 20;

string names[SIZE];

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

names[i] = "Sirius Black";

}

1. The arrays array1 and array2 each hold 25 integer elements. Write code that copies the values in array1 to array2.

int array1[25], array2[25];

for(int i = 0; i < 25; i++){

array2[i] = array1[i];

}

1. In a program you need to store the identification numbers of 10 employees (as ints) and their weekly gross pay (as doubles).  
   1. Define two arrays that may be used in parallel to store the 10 employee identification numbers and 10 weekly gross pay amounts
   2. Write a loop that uses these arrays to print each employee’s identification number and weekly gross pay.

int emp\_ID[10];

double emp\_pay[10];

for (int i=0; i < 10; i++)

{

cout << "Employee ID: "<<emp\_ID[i] << endl;

cout << "Gross Pay: " <<emp\_pay[i] << endl;

}

1. Rewrite the code for Problem 8 to define and use an array of Payroll structures instead of two parallel arrays. A Payroll structure should hold an employee ID and weekly gross pay amount.

struct Payroll = {

int emp\_ID;

double emp\_pay;

};

Payroll employees[10];

for (int i=0; i < 10; i++) {

cout << "Employee ID: "<< employees[i].emp\_ID << endl;

cout << "Gross Pay: " << employees[i].emp\_pay << endl;

}

1. Write a function that accepts the above Payroll structure and size as parameters, calculates total pay for all employees and returns the total.

double sumPayroll(Payroll employees[], *const* int SIZE){

double total = 0;

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

total += employees[i].emp\_pay;

}

*return* total;

}

1. class Rectangle

{

private:

int length, width;

public:

Rectangle()

{length = 0; width = 0;}

void setLength(int l)

{length = l;}

void setWidth (int w)

{width = w;}

};

Given the above class Rectangle, write statements to create an array of 5 rectangles and store a length of 10 and a width of 5 to the third element in the array.

Rectangle array[5];

array[2].setLength(10);

array[2].setWidth(5);

1. Why is selection sort more efficient than bubble sort on large arrays

*// Bubble sort normally has to make many data exchanges to place a value in its correct position. Selection*

*// sort determines which value belongs in the position currently being filled with the correctly ordered next*

*// value and then places that value directly there.*

*// Selection sort performs a smaller number of swaps compared to bubble sort; therefore, even though both sorting methods are of O(N2), selection sort performs faster and more efficiently!*

1. Assume that empName and empID are two parallel arrays of size numEmp that hold employee data. Using a bubble sort, write the code that will sort the empID array in ascending ID number order, such that the two arrays remain parallel. That is, after sorting, for all indexes in the arrays, empName[index] must still be the name of the employee whose ID is in empID [index].

*const* int numEmp = 5;

string empName[numEmp] = {"John", "Sam", "Mary", "Anna", "Ali"};

int empID[numEmp] = {17, 320, 444, 890, 101};

bool swap;

long temp;

string tempstr;

do

{

swap = false; *// There have been no swaps yet on this pass*

for (int count = 0; count < (numEmp - 1); count++)

{

if (empID[count] > empID[count + 1])

{

temp = empID[count];

empID[count] = empID[count + 1];

empID[count + 1] = temp;

tempstr = empName[count];

empName[count] = empName[count + 1];

empName[count + 1] = tempstr;

swap = true;

}

}

} while (swap); *// While there was a swap on the previous pass*

1. Assume an array exists containing 10 unsorted 3 digit account numbers. Write code to perform a linear search for account number 345.

int linearSearch(*const* int array[], int size, int value){

int index = 0; *// Where we start - increase to loop through array*

int position = -1; *// Where we will stored the record of position if found else return -1*

bool found = false; *// Flag to indicate if value was found;*

while(index < size && !found ){ *// While index is less than the size of array AND found is false - loop*

if(array[index] == value){ *// If the value is found*

found = true; *// Set the flag*

position = index; *// Set the position to the index it was found at*

}

index++;

}

*return* position;

}

1. Given the array definition:

const int numbers [SIZE] = {18, 17, 12, 14};

and we want to pass the array to the function processArray in the following manner:

processArray (numbers, SIZE );

Write the correct function header for the processArray function.

Answer: void processArray(const int array[], int size)

1. Car Class  
   Write the C++ code for a class named Car that has the following member variables:

year. An int that holds the car’s model year.  
make. A string object that holds the make of the car  
speed. An int that holds the car’s current speed  
  
In addition, the class should have the following member functions  
  
Constructor. The constructor should accepts the car’s year and make as arguments and assign these values to the object’s year and make member variables. The constructor should initialize the speed member variable to 0.  
  
Accessors. Appropriate accessor functions should be created to allow values to be retrieved from an object’s year, make, and speed member variables.  
  
accelerate. The accelerate function should add 5 to the speed member variable each time it is called.

Brake. The brake function should subtract 5 from the speed member variable each time it is called.

#include <iostream>

#include <string>

using namespace std;

*// Car class declaration*

class Car

{

private:

int year; *// Model year of the Car*

string make; *// Make of the Car*

int speed; *// Current speed of the Car (mph)*

public:

*// Constructor with default parameters*

Car(int y = 2013, string m = "Unknown")

{

year = y;

make = m;

speed = 0;

}

*// Accessors (i.e. "get" functions)*

int getYear()

{

*return* year;

}

string getMake()

{

*return* make;

}

int getSpeed()

{

*return* speed;

}

*// Mutators*

void accelerate()

{

speed += 5;

}

void brake()

{

if (speed >= 5)

speed -= 5;

else

speed = 0;

}

};

*// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* main \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

int main()

{

Car hotRod(2013, "Mazda"); *// Create a Car object*

*// Describe the car (This line was not required by the problem specs.)*

cout << "I'm in my " << hotRod.getYear() << " "

<< hotRod.getMake() << " hot rod.\n\n";

*// Accelerate*

cout << "I'm accelerating ... \n\n";

for (int faster = 1; faster <= 5; faster++)

{

hotRod.accelerate();

cout << "Current speed: " << hotRod.getSpeed() << " mph. \n";

}

*// Now stop*

cout << "\nNow I'm braking ... \n\n";

for (int slower = 1; slower <= 5; slower++)

{

hotRod.brake();

cout << "Current speed: " << hotRod.getSpeed() << " mph. \n";

}

*return* 0;

}

1. Find the errors:
   1. struct

{ int x;

double y;

};

Answer: Missing variable name in declaration

* 1. struct Values

{ string name;

int age;

}

Answer: No closing semi-colon

1. The average number of comparisons performed by linear search to find an item in an array of N elements is \_\_\_\_\_\_ N/2\_\_\_\_\_\_\_\_\_.
2. In a binary search, after three comparisons have been made, only \_\_1/8th\_\_ of the array will be left to search.
3. To sort N numbers, bubble sort continues making passes through the array until \_\_\_\_ \_there were no exchanges on the previous pass\_\_\_\_\_\_\_\_\_\_\_\_