1.1 Because the computer can be programmed to do so many different tasks.

1.2 (CPU), main memory, secondary storage devices, input devices, output devices.

1.3 Arithmetic and Logic Unit (ALU), and Control Unit

1.4 Fetch: The CPU’s control unit fetches the program’s next instruction from main memory.

Decode: The control unit decodes the instruction, which is encoded in the form of a number. An electrical signal is generated.

Execute: The signal is routed to the appropriate component of the computer, which causes a device to perform an operation.

1.5 a number assigned to each section of memory.

1.6 Program instructions and data are stored in main memory while the program is operating. Main memory loses its contents when power is removed . Secondary storage holds data for long periods of time

1.7 Operating Systems and Application Software

1.8 The operating system

1.9 A utility program

1.10 application software

1.11 A set of steps for performing a task or solving a problem.

1.12 To ease the task of programming.

1.13 A low-level language is close to the level of the computer, and resembles the system’s numeric machine language. A high-level language is closer to the level of human readability, and resemble natural languages.

1.14 That a program may be written on one type of computer and run on another type.

1.15 The preprocessor reads the source file searching for commands that begin with the # symbol. These are commands that cause the preprocessor to modify the source file in some way. The compiler translates each source code instruction into the appropriate machine language instruction, and creates an object file. The linker combines the object file with necessary library routines.

1.16 Source file: contains program statements written by the programmer.

Object file: machine language instructions,

Executable file: code ready to run on the computer. Includes the machine language from an object file, and the necessary code from library routines.

1.17 A text editor, compiler, debugger, and other utilities, integrated into one package.

1.18 A key word has a special purpose, and is defined as part of a programming language. A programmer-defined identifier is a word or name defined by the programmer.

1.19 Operators perform operations on one or more operands. Punctuation symbols mark the beginning or ending of a statement

1.20 A line is a single line as it appears in the body of a program. A statement is a complete instruction that causes the computer to perform an action.

1.21 Because their contents may be changed.

1.22 The original value is overwritten.

1.23 The variable must be defined.

1.24 Input, processing, and output.

1.25 The program’s purpose, information to be input, the processing to take place, and the desired output.

1.26 To imagine what the computer screen looks like while the program is running. This helps define input and output.

1.27 A chart that depicts each logical step of the program in a hierarchical fashion.

1.28 The programmer steps through each statement in the program from beginning to end. The contents of variables are recorded, and screen output is sketched.

1.29 It translates each source code statement into the appropriate machine language statement..

1.30 A logical error that occurs while the program is running.

1.31 By the compiler

1.32 To determine if a logical error is present in the program.

1.33 Procedural programs are made of procedures, or functions. Object-oriented programs are centered on objects, which contain both data and the procedures that operate on the data.

2.1 // A crazy mixed up program #include <iostream> using namespace std;

int main() { cout << "In 1492 Columbus sailed the ocean blue."; return 0;

}

2.2 // It's a mad, mad program #include <iostream> using namespace std;

int main() { cout << "Success\n"; cout << "Success"; cout << " Success\n\n"; cout << "Sucess\n"; return 0; }

2.3 The works of Wolfgang include the following The Turkish March and Symphony No. 40 in G minor.

2.4 // Today's Date: September 3, 2012 #include <iostream> using namespace std;

int main() { cout << "Teresa Jones\n"; cout << "127 West 423rd Street\n"; cout << "San Antonio, TX 55555\n"; cout << "555-555-1212\n"; return 0;

}

2.5 Variables: little and big.

Constants: 2, 2000, “The little number is ”, “The big number is ”

2.6 The value is number

2.7 99bottles: Variable names cannot begin with a number.

r&d: Variable names may only use alphabetic letters, digits, or underscores

2.8 No. Variable names are case sensitive.

2.9 A) short, or unsigned short.

1. int
2. They both use the same amount of memory.
   1. They both use the same amount of memory.
   2. 67, 70, 87
   3. ‘B’
   4. ‘Q’ uses one byte

“Q” uses two bytes

“Sales” uses six bytes

‘\n’ uses one byte

* 1. #include <iostream> using namespace std;

int main() { char first, middle, last; first = 'T'; middle = 'E'; last = 'G'; cout << first << " " << middle << " " << last << endl; return 0;

}

* 1. The string constant “Z” is being stored in the character variable letter.
  2. The string header file 2.17 #include <iostream> #include <string> using namespace std;

int main() {

string name = "John Smith"; string address = "224 Maple Street\nClyde, NC 28721"; string phone = "555-5050";

cout << name << endl; cout << address << endl; cout << phone << endl << endl; return 0; }

* 1. No
  2. 6.31E17
  3. #include <iostream> using namespace std;

int main() { int age; float weight;

age = 26; weight = 180; cout << "My age is " << age << endl; cout << "My weight is " << weight << endl; return 0;

}

* 1. Invalid. The value on the left of the = operator must be an lvalue.
  2. int x = 7, y = 16, z = 28;
  3. The variable number is assigned a value before it is defined. Correct the program by moving the statement number = 62.7; to the point after the variable declaration. Here is the corrected program:

#include <iostream> using namespace std;

int main() { double number; number = 62.7; cout << number << endl; return 0;

}

* 1. Integer division. The value 23 will be stored in portion.
  2. const float E = 2.71828;   
     const int MINUTES\_IN\_A\_YEAR = 5.256E5;   
     const float G\_FEET = 32.2;  
      const float G\_METERS = 9.8;   
     const int METERS\_IN\_A\_MILE = 1609
  3. iostream
  4. True
  5. B
  6. cin >> miles >> feet >> inches;
  7. Include one or more cout statements explaining what values the user should enter.
  8. #include <iostream> using namespace std;

int main() { double pounds, kilograms;

cout << "Enter your weight in pounds: ";   
cin >> pounds; // The following line does the conversion.   
// One kilogram weighs 2.2 pounds.

kilograms = pounds / 2.2;  
 cout << "Your weight in kilograms is ";  
 cout << kilograms << endl;   
return 0;

}

3.7 *Value*

21

2

31

5

24

2

69

0

30

3.8 y = 6 \* x; a = 2 \* b + 4 \* c; y = x \* x; or y = pow(x, 2.0); g = (x + 2) / (z \* z); or g = (x + 2.0) / pow(z, 2.0); y = (x \* x) / (z \* z); or y = pow(x, 2.0) / pow (z, 2.0);

3.9 *If the user enters… The program displays…*

2 6

1. 27

4.3 20.49

1. 38
   1. #include <iostream> #include <cmath>

using namespace std;

int main() { double volume, radius, height;

cout << "This program will tell you the volume of\n";   
 cout << "a cylinder-shaped fuel tank.\n"; cout << "How tall is the tank? "; cin >> height;   
  
 cout << "What is the radius of the tank? "; cin >> radius;   
 volume = 3.14159 \* pow(radius, 2.0) \* height; cout << "The volume of the tank is " << volume << endl; return 0; }

* 1. A) 2 B) 17.0

1. 2.0
2. 2.4
3. 2.4
4. 2.4
5. 4
6. 27
7. 30
8. 27.0
   1. #include <iostream> using namespace std;

int main() { char letter;

cout << "Enter a character: ":

cin >> letter; cout << "The ASCII code for " << letter;   
cout << " is " << static\_cast<int>(letter) << endl;   
return 0;

}

* 1. 9

9.5

9

* 1. total = subtotal = tax = shipping = 0;
  2. A) x += 6; B) amount -= 4; C) y \*= 4;

1. total /= 27;
2. x %= 7;
3. x += (y \* 5);
4. total -= (discount \* 4);
5. increase \*= (salesRep \* 5);
6. profit /= (shares – 1000);
   1. 3

11

1

* 1. A) cout << setw(9) << fixed << setprecision(2) << 34.789;

1. cout << setw(5) << fixed << setprecision(3) << 7.0;
2. cout << fixed << 5.789e12;
3. cout << left << setw(7) << 67;
   1. #include <iostream> #include <string> using namespace std;

int main() { string person = "Wolfgang Smith"; cout << right; cout << setw(20); cout << person << endl; cout << left; cout << person << endl; return 0; }

* 1. #include <iostream> #include <iomanip> using namespace std;

int main() { const double PI = 3.14159; double degrees, radians;

cout << "Enter an angle in degrees and I will convert it\n"; cout << "to radians for you: "; cin >> degrees; radians = degrees \* PI / 180; cout << degrees << " degrees is equal to "; cout << setw(5) << left << fixed << showpoint << setprecision(4) << radians << " radians.\n"; return 0;

}

* 1. cos: Returns the cosine of the argument.

exp: Returns the exponential function of the argument.

fmod: Returns the remainder of the first argument divided by the second argument. log: Returns the natural logarithm of the argument. log10: Returns the base-10 logarithm of the argument.

pow: Returns the value of the first argument raised to the power of the second argument.

sin: Returns the sine of the argument.

sqrt: Returns the square root of the argument. tan: Returns the tangent of the argument.

* 1. x = sin(angle1) + cos(angle2);
  2. y = pow(x, 0.2); // 0.2 is equal to 1/5
  3. y = 1 / sin(a);