|  |
| --- |
| 1. Run the program and determine the smallest value of n that causes failure. Try changing the type of i to short and running the program Then try long. From these experiments, what can you conclude about the number of bits used to store integer types on your machine?     int type: n is 46,341 (46,341 \* 46,341 = 2,147,488,281 = 1000 0000 0000 0000 0001 0010 0001 1001), 32-bit    short type: n is 182 (182 \* 182 = 33,124 = 1000 0001 0110 0100), 2-bit    long type: n is 3,037,000,500 (3,037,000,500 \* 3,037,000,500 = 9,223,372,037,000,250,000 = 1000 0000 0000 0000 0000 0000 0000 0000 0000 1000 1010 1011 1100 0010 1001 0000), 8-bit |
| 1. Modify the square2.c program of Section 6.3 so that it pauses after every 24 squares and displays the following message:   1 #include <stdio.h>  2  3 int main(void)  4 {  5 int i, n;  6  7 printf("This program prints a table of squares. \n");  8 printf("Enter number of entries in table: ");  9 scanf("%d", &n);  10  11 while (getchar() != '\n'); //Remove white-space  12  13 for (i = 1; i <= n; i++) {  14 printf("%10d%10d\n", i, i \* i);  15  16 if (i % 24 == 0) {  17 for (;;) {  18 printf("Press Enter to continue... \n");  19 if (getchar() != '\n')  20 while (getchar() != '\n');  21 else  22 break;  23 }  24 }  25 }  26  27 return 0;  28 } |
| 1. Modify the sum2.c program of Section 7.1 to sum a series of double values.   1 #include <stdio.h>  2  3 int main(void)  4 {  5 double n, sum = 0.0;  6  7 printf("This program sums a series of float-point. \n");  8 printf("Enter float-point (0 to terminate): ");  9  10 scanf("%lf", &n);  11 while (n != 0) {  12 sum += n;  13 scanf("%lf", &n);  14 }  15  16 printf("The sum is: %g \n", sum);  17  18 return 0;  19 } |
| 1. Write a program that translates an alphabetic phone number into numeric form:   1 #include <stdio.h>  2  3 int main(void)  4 {  5 char ch;  6  7 printf("Enter phone number: ");  8 while ( (ch = getchar()) != '\n' )  9 {  10 switch(ch)  11 {  12 case 'A': case 'B': case 'C':  13 putchar('2');  14 break;  15  16 case 'D': case 'E': case 'F':  17 putchar('3');  18 break;  19  20 case 'G': case 'H': case 'I':  21 putchar('4');  22 break;  23  24 case 'J': case 'K': case 'L':  25 putchar('5');  26 break;  27  28 case 'M': case 'N': case 'O':  29 putchar('6');  30 break;  31  32 case 'P': case 'R': case 'S':  33 putchar('7');  34 break;  35  36 case 'T': case 'U': case 'V':  37 putchar('8');  38 break;  39  40 case 'W': case 'X': case 'Y':  41 putchar('9');  42 break;  43  44 default:  45 putchar(ch);  46 break;  47 }  48 } putchar('\n');  49  50 return 0;  51 } |
| 1. Write a program that computes the value of a word by summing the values of its letters:   1 #include <stdio.h>  2 #include <ctype.h>  3  4 int main(void)  5 {  6 int value;  7 char ch;  8  9 printf("Enter a word: ");  10 while ( (ch = getchar()) != '\n')  11 {  12 switch ( toupper(ch))  13 {  14 case 'A': case 'E': case 'I': case 'L': case 'N':  15 case 'O': case 'R': case 'S': case 'T': case 'U':  16 value += 1;  17 break;  18  19 case 'D': case 'G':  20 value += 2;  21 break;  22  23 case 'B': case 'C': case 'M': case 'P':  24 value += 3;  25 break;  26  27 case 'F': case 'H': case 'V': case 'W': case 'Y':  28 value += 4;  29 break;  30  31 case 'K':  32 value += 5;  33 break;  34  35 case 'J': case 'X':  36 value += 8;  37 break;  38  39 case 'Q': case 'Z':  40 value += 10;  41 break;  42 }  43 }  44  45 printf("Scrabble value: %d \n", value);  46  47 return 0;  48 } |
| 1. Write a program that prints the values of ...   1 #include <stdio.h>  2  3 int main(void)  4 {  5 printf("sizeof(int): %zu \n", sizeof(int));  6 printf("sizeof(short): %zu \n", sizeof(short));  7 printf("sizeof(long): %zu \n", sizeof(long));  8 printf("sizeof(float): %zu \n", sizeof(float));  9 printf("sizeof(double): %zu \n", sizeof(double));  10 printf("sizeof(long double): %zu \n", sizeof(long double)); 11  12 return 0;  13 } |
| 1. Modify Programming Project 6 from Chapter 3 so that the user may add, subtract, multiply, or divide two fractions (by entering either +, -, \* or / between the fractions).   1 #include <stdio.h>  2  3 int main(void)  4 {  5 int num1, denom1;  6 int num2, denom2;  7 int operator;  8 int result\_num, result\_denom;  9  10 printf("Enter two fractions separated by a plus sign: ");  11 scanf("%d /%d ", &num1, &denom1);  12 operator = getchar();  13 scanf("%d /%d", &num2, &denom2);  14  15 switch (operator)  16 {  17 case '+':  18 result\_num = num1 \* denom2 + num2 \* denom1;  19 result\_denom = denom1 \* denom2;  20 break;  21  22 case '-':  23 result\_num = num1 \* denom2 - num2 \* denom1;  24 result\_denom = denom1 \* denom2;  25 break;  26  27 case '\*':  28 result\_num = num1 \* num2;  29 result\_denom = denom1 \* denom2;  30 break;  31  32 case '/':  33 result\_num = num1 \* denom2;  34 result\_denom = denom1 \* num2;  35 break;  36 }  37  38 printf("The sum is %d / %d \n", result\_num, result\_denom);  39  40 return 0;  41 } |
| 1. Modify Programming Project 8 from Chapter 5 so that the user enters a time using the 12-hour clock.   1 #include <stdio.h>  2 #include <ctype.h>  3  4 int main(void)  5 {  6 int hour, minute, convert;  7 char is\_24hour;  8 printf("Enter a 12-hour time: ");  9 scanf("%d:%d", &hour, &minute);  10 scanf(" %c", &is\_24hour);  11  12 while (getchar() != '\n');  13  14 if (toupper(is\_24hour) == 'P') {  15 convert = (hour + 12) \* 60 + minute;  16 } else if(toupper(is\_24hour) == 'A') {  17 convert = hour \* 60 + minute;  18 }  19  20 if (convert < (8 \* 60) + 0) {  21 hour = 8; minute = 0;  22 convert = (10 \* 60) + 16;  23  24 } else if (convert < (9 \* 60) + 43) {  25 hour = 9; minute = 43;  26 convert = (11 \* 60) + 52;  27  28 } else if (convert < (11 \* 60) + 19) {  29 hour = 11; minute = 19;  30 convert = (13 \* 60) + 31;  31  32 } else if (convert < (12 \* 60) + 47) {  33 hour = 12; minute = 47;  34 convert = (15 \* 60) + 0;  35  36 } else if (convert < (14 \* 60) + 0) {  37 hour = 14; minute = 0;  38 convert = (16 \* 60) + 8;  39  40 } else if (convert < (15 \* 60) + 45) {  41 hour = 15; minute = 45;  42 convert = (17 \* 60) + 55;  43  44 } else if (convert < (19 \* 60) + 0) {  45 hour = 19; minute = 0;  46 convert = (21 \* 60) + 20;  47  48 } else if (convert < (21 \* 60) + 45) {  49 hour = 21; minute = 45;  50 convert = (23 \* 60) + 58;  51  52 } else {  53 hour = minute = -1;  54 }  55  56 if (hour == -1) {  57 printf("You missed all the flight in the day \n");  58 } else {  59 printf("Closest departure time is %d:%.2d %s, ",  60 hour, minute, (hour >= 12) ? ("a.m.") : ("p.m.")  61 );  62  63 printf("arriving at %d:%.2d %s \n",  64 convert / 60, convert % 60, ((convert / 60) >= 12) ? (" a.m.") : ("p.m.")  65 );  66 }  67  68 return 0;  69 } |
| 1. Write a Program that asks the user for a 12-hour time, then displays the time in 24-hour form:   1 #include <stdio.h>  2 #include <ctype.h>  3  4 int main(void)  5 {  6 int hour, minute, convert;  7 char is\_24hour;  8  9 printf("Enter a 12-hour time: ");  10 scanf("%d:%d", &hour, &minute);  11 scanf(" %c", &is\_24hour);  12  13 while (getchar() != '\n');  14  15 if (toupper(is\_24hour) == 'P') {  16 hour = 12 + hour;  17 } else if(toupper(is\_24hour) == 'A') {  18 hour = hour;  19 }  20  21 printf("Equivalent 24-hour time: %d:%d \n", hour, minute);  22  23 return 0;  24 } |
| 1. Write a program that counts the number of vowels (*a*, *e*, *i*, *o* and *u*) in a sentence:   1 #include <stdio.h>  2 #include <ctype.h>  3  4 int main(void)  5 {  6 int ch;  7 int counts;  8  9 counts = 0;  10  11 printf("Enter a sentence: ");  12 while ( (ch = getchar()) != '\n') {  13 switch ( toupper(ch) )  14 case 'A': case 'E':  15 case 'I': case 'O': case 'U':  16 counts++;  17 }  18  19 printf("Your sentence contains %d vowels \n", counts);  20  21 return 0;  22 } |
| 1. Write a program that takes a first name and last name entered by the user and displays the last name, a comma, and the first initial, followed by a period:   1 #include <stdio.h>  2 #include <ctype.h>  3  4 int main(void)  5 {  6 char first\_name, last\_name;  7 char ch;  8  9 printf("Enter a first and last name: ");  10  11 scanf(" %c", &first\_name);  12  13 // skipping a remained first-name.  14 while ( isalpha( ch = getchar() ) ) ;  15  16 //  17 scanf(" %c", &last\_name);  18 putchar(last\_name);  19  20 // printing a last-name  21 while ( isalpha( last\_name = getchar() ) )  22 putchar(last\_name);  23  24 printf(", %c \n", first\_name);  25  26 return 0;  27 } |
| 1. Write a program that evaluates an expression:   1 #include <stdio.h>  2  3 int main(void)  4 {  5 double result, operand;  6 char operator;  7  8 printf("Enter an expression: ");  9 scanf(" %lf", &result);  10 while (1) {  11 for (;;) {  12 operator = getchar();  13  14 if (operator == '+' || operator == '-'  15 || operator == '\*' || operator == '/'  16 || operator == '\n')  17 break;  18 }  19  20 if (operator == '\n') break;  21  22 scanf(" %lf", &operand);  23  24 switch (operator) {  25 case '+':  26 result += operand;  27 break;  28  29 case '-':  30 result -= operand;  31 break;  32  33 case '\*':  34 result \*= operand;  35 break;  36  37 case '/':  38 result /= operand;  39 break;  40 }  41 }  42  43 printf("Value of expression: %g \n", result);  44  45 return 0;  46 } |
| 1. Write a program that calculates the average word length for a sentence:   1 #include <stdio.h>  2  3 int main(void)  4 {  5 char ch;  6 int count\_word;  7 int count\_alphabet;  8  9 count\_word = count\_alphabet = 0;  10  11 printf("Enter a sentence: ");  12 do {  13 ch = getchar();  14  15 if (ch == ' ') count\_word++;  16 else count\_alphabet++;  17 } while (ch !='\n');  18  19 printf("Average word length: %.1f \n",  20 (double)(count\_alphabet - 1) / (count\_word + 1));  21 //exclude '\n' character, include remaining word.  22  23 return 0;  24 } |
| 1. Write a program that uses Newton’s method to compute the square root of a positive floating-point number:   1 #include <stdio.h>  2 #include <math.h>  3  4 int main(void)  5 {  6 double x, y, average;  7  8 printf("Enter a positive number: ");  9 scanf("%lf", &x);  10  11 y = 1.0;  12 while (1) {  13 average = (y + x/y) / 2.0;  14 if ( fabs(y - average) < 0.00001 )  15 break;  16 else  17 y = average;  18 }  19 printf("Square root: %g \n", average);  20  21 return 0;  22 } |
| 1. Write a program that computes the factorial of a positive integer:   1 #include <stdio.h>  2 #include <math.h>  3  4 int main(void)  5 {  6 unsigned int n;  7 unsigned long long factorial;  8  9 printf("Enter a positive integer: ");  10 scanf("%d", &n);  11  12 factorial = 1;  13 while (n > 0)  14 factorial \*= n--;  15  16 printf("result: %llu \n", factorial);  17  18 return 0;  19 } |