**TUTORIAL 4**

**Question 1**

Write a function that inputs a series of 10 integers and passes them one at a time to function *isEven*, which uses the remainder operator to determine whether an integer is even. The function should take an integer argument and return 1 if the integer is even and 0 otherwise.

**Question 2**

Write a function that displays a solid rectangle of asterisks whose sides are specified in the integer parameters *side1* and *side2.* For example, if the sides are 4 and 5, the function displays the following:



**Question 3**

Write a function that returns the largest of four floating-point numbers.

**Question 4**

Write a function *Distance* that calculates the distance between two points (x1, y1) and (x2, y2). All numbers and return values should be of type double.

**Question 5**

A parking garage charges a $2.00 minimum fee to park for up to three hours and an additional $0.50 per hour for each hour or part thereof over three hours. The maximum charge for any given 24-hour period is $10.00. Assume that no car parks for longer than 24 hours at a time. Write a program that will calculate and print the parking charges for each of three customers who parked their cars in this garage yesterday. You should enter the hours parked for each customer. Your program should print the results in a tabular format, and should calculate and print the total of yesterday's receipts. The program should use the function *calculateCharges* to determine the charge for each customer. Your outputs should appear in the following format:

*Car Hours Charge*

*1 1.5 2.00*

*2 4.0 2.50*

*3 24.0 10.00*

*TOTAL 29.5 14.50*

**Question 6**

Write a C program that plays the game of “guess the number” as follows: Your program chooses the number to be guessed by selecting an integer at random in the range 1 to 1000. The program then types:

*I have a number between 1 and 1000.*

*Can you guess my number?*

*Please type your first guess.*

The player then types a first guess. The program responds with one of the following:

1. *Excellent! You guessed the number!*

*Would you like to play again (Press 1 to continue and 0 to exit)?*

1. *Too low. Try again.*
2. *Too high. Try again.*

If the player’s guess is incorrect, your program should loop until the player finally gets the number right. Your program should keep telling the player *Too high* or *Too low* to help the player reach the correct answer. Your program should use the function *Decision* output the appropriate response to the value input.

[Hint: use the function *rand() % (max\_number + 1 - minimum\_number) + minimum\_number* to obtain a random number]