CS 1336.007 - Programming Fundamentals
Tuesdays & Thursdays 1 pm to 2:15 pm ECSS 2.312

Instructor	Dr. Jey Veerasamy Office: ECSS 3.231 Office Phone: 972-883-4241 (you can call me during office hours) E-Mail: jeyv@utdallas.edu Office hours: Tuesdays 10:30 am - 12:30 pm Thursdays 10:30 am - 12:30 pm
TA	TBD

Prequisites: None

Co-requisites:

CS 1136 – A sequence of labs will be assigned and graded for CS 1136, these are separate from the assignments made in CS 1336. Students earn separate grades for CS 1336 and CS 1136.

Catalog Description:

CS 1336 Programming Fundamentals
Introduction to computers. Primitive data types, variable declarations, variable scope, and primitive operations. Control statements. Methods/functions. Arrays, and strings using primitive data arrays. Output formatting. Debugging techniques.

Designed for students with no prior computer programming experience. This class cannot be used to fulfill degree requirements for majors in the School of Engineering and Computer Science.

Learning Outcomes:

After successful completion of this course, the student should be able to:

- Ability to develop algorithmic solutions for use on computers
- Ability to perform console input and output, utilize basic operators, and perform sequential processing
- Ability to utilize the basic control structures for selection
- Ability to utilize the basic control structures for repetition logic
- Ability to perform sequential file input and output
- Ability to develop programs in a functional form
- Ability to process data in arrays

Textbook:

Starting Out with C++, From Control Structures through Objects (8th edition); Gaddis, Tony; Addison-Wesley Publishing. ISBN 978-0-13-376939-5

Notes regarding textbook material:

As you read the text watch the corresponding VideoNotes. The VideoNotes are available at http://www.pearsonhighered.com/gaddis/.

Additional optional resources:

Programming using JavaScript: www.khanacademy.org/cs, www.utdallas.edu/~jeyv/kajs
C++ language tutorial www.cplusplus.com/files/tutorial.pdf
C++ reference: www.cppreference.com
C++ tutorial www.learncpp.com

Academic Calendar: See "Coures Homepage" witnin elearning for the detailed schedule. It will be updated with lecture notes as the semester proceeds.

Course & Instructor Policies:

The final grade will be computed as follows:

Tests	50%	*2 tests contributing 25% each* All tests are closed book and closed notes & they will be conducted in the computers @ the testing center. If the testing center is not available, paper test will be given. Tests will focus more on concepts, but coding will be required. Tests may include a few multiple-choice or fill-in-the-blank style questions as well. Necessary documentation will be provided to avoid the need for memorization as much as possible. All make-up tests are scheduled during the week following the actual test date at the discretion of the instructor - advance notice is required.
Assingments	40%	*There will be several assignments distributed throughout the course, due every week or every other week. All of them will have equal weightage.* I encourage everyone to submit the assignments 1 or 2 days early. Do not wait until the last minute to submit it. But I do understand things happen and occasionally you may not be able to submit assignments on time. My policy is to assess 1% penalty for every 2 hours. For example, if you submit the assignment exactly 1 day later, 12% penalty will be assessed. Late assignments will be accepted up to 4 days. You won't be able to submit it after 4 days and your assignment grade will be set to 0. You can ask for clarifications and help in the Assignments forum. If you need help with your code, it is ok to post 1 or 2 lines of code, but do not post your full program - email it to TA or professor instead. You are expected to start working on them as soon as they are posted. Do not expect us to rescue you on the day of submission.
Pair programming project	10%	As we get into middle of the term, whole class will be divided into teams of 2 students each by the instructor. Then each team will select a problem from a big list, discuss, work together and come up with the solution in the remaining term. Each team should document their interactions & submit it towards the end of the term. Your team work is more important compared to the final output.

Course credit is only given for work assigned in the course schedule. No extra work will be assigned nor will extra credit be given for any extra work performed by a student. Instructor

is responsible for grading all the tests, quizzes & weekly participation. TA will be responsible for grading assignments and weekly activities. So, contact the TA directly for any grading issues related to those items. If you cannot resolve it with TA, bring it to instructor's attention.

In addition to meeting the instructor before or after the class, you can also visit the instructor or TA during respective office hours. You can call instructor's office phone during office hours as well. However, be prepared to hold and wait if the instructor is busy with another student in the office. Additionally, you are welcome to email the instructor or TA. If you need help with your program, in addition to problem description & applicable error messages, include your source files with your email too, so that we can review & help you efficiently.

Letter grades will be assigned as follows:

97-100	A+	94-97	A	90-94	A-
87-90	B+	84-87	B	80-84	B-
77-80	C+	74-77	C	70-74	C-
67-70	D+	64-67	D	60-64	D-
Below 60	F				

Programming Assignments:

Each programming assignment will require the students to spend a few hours to even days programming in a computer. Right way to approach the programming assignments is to start on them at least one week earlier than the due date & get help when you get stuck (you can approach the instructor, TA, or tutors at CS tutoring lab for help). Do not waste lots of hours trying to fix a small glitch. In simple words, your approach will determine whether programming assignments provide an enjoyable learning experience or end up like a painful useless activity.

Programming assignments will be graded on a 100 point basis, utilizing the following criteria:

		Max Score
Source Code	Overall design	40%
	Formatting	10%
	Naming	10%
	Capitalization	10%
Execution	Nominal cases	25%
	Special cases	5%
Total		**100%**

Programming assignments should be turned in by means of eLearning. You need to submit only .cpp files for individual assignments, unless explicitly stated otherwise.

Any standard C++ compiler and Integrated Development Environment (IDE) can be used to develop, debug and run your programs. Microsoft Visual Studio, Microsoft Visual Express, Code::Blocks, NetBeans, Eclipse and jGRASP are a few popular tools. More information about these tools will be provided in elearning.

Here are a few assignments from previous course & they have been included here for reference. Official ones for this course will be posted in elearning.

Assignment 1: Write a program to perform check-out functionality for a simple Farmers Market store with exactly 5 products:

Product	Price per pound
Bananas	$ 0.44
Apples	$ 0.99
Cucumbers	$ 1.19
Carrots	$ 0.89
Oranges	$ 0.79

After getting the weight for each product purchase and compute & output the total purchase amount in the end. Do not use loops or arrays in this assignment. You can assume that all the user inputs are valid & the user will enter 0 for products (s)he does not purchase – no need to do any input validation explicitly. Use meaningful variable names & achieve "self-documenting" style of coding.

Assignment 2: We will expand Program 1 using decision structure to apply discounts. Farmers market introduces two types of discounts to attract more customers and make each customer to purchase more.

1. Discount card program to attract customers. All customers who have signed up for it are considered as "special customers" and they get automatic minimum 10% discount off based on the total purchase amount. However, this cannot be combined with other discounts.
2. All customers are eligible for the following discount program. If the total purchase is above $50, 5% discount will apply. If the total purchase exceeds $75, 10% discount will apply. If the total purchase exceeds $100, 15% discount will apply.

After computing the total purchase amount, program should determine the discount & output the following items if a discount is applied:

- discount % and discount amount
- discounted total

Assignment 3: We will implement two versions of guessing game in this assignment.

Version 1: You think of a number, computer will ask the questions and determine your number. Start the game by asking the user for the range: low number & high number. Then, ask the user to think of a number within the specified range. Then, the program should ask you a series of questions and determine your guess based on your answers. Name this program as GuessingGame1.cpp

Sample run 1:

Enter the low end value for the range: 1
Enter the high end value for the range: 100
Guess a number between 1 and 100 (both inclusive)
and get ready to answer a few questions.
How about 50 (<,=,>)? <
How about 25 (<,=,>)? <
How about 12 (<,=,>)? >
How about 18 (<,=,>)? >
How about 21 (<,=,>)? <
How about 19 (<,=,>)? >
Your guess is 20
It was a good game!

Sample run 2:

Enter the low end value for the range: 1
Enter the high end value for the range: 100
Guess a number between 1 and 100 (both inclusive)
and get ready to answer a few questions.
How about 50 (<,=,>)? >
How about 75 (<,=,>)? <
How about 62 (<,=,>)? >
How about 68 (<,=,>)? >
How about 71 (<,=,>)? =
It was a good game!

Sample run 3:

Enter the low end value for the range: 101
Enter the high end value for the range: 200
Guess a number between 101 and 200 (both inclusive)
and get ready to answer a few questions.
How about 150 (<,=,>)? <
How about 125 (<,=,>)? <
How about 112 (<,=,>)? =
It was a good game!

Sample run 4:

Enter the low end value for the range: 201
Enter the high end value for the range: 400
Guess a number between 201 and 300 (both inclusive)
and get ready to answer a few questions.
How about 300 (<,=,>)? <
How about 250 (<,=,>)? =
It was a good game!

Version 2: Computer guesses the number, you will ask a series of questions to the computer until you come up with the number computer guessed. Name this program as GuessingGame2.cpp.

When you submit the assignment, remember to attach both .cpp files. Have fun!

Sample run 1:

Enter the low end value for the range: 201
Enter the high end value for the range: 400
Ok, I have chosen a number & you can start to guess!
350
too high
250
too low
340
too low
348
Great. It was a good game!

Sample run 2:

Enter the low end value for the range: 950
Enter the high end value for the range: 1050
Ok, I have chosen a number & you can start to guess!
999
too high
969
too low
977
too high
975
Great. It was a good game!

Assignment 4: Connect4 game

Go to http://www.mathsisfun.com/games/connect4.html and play the Connect4 game few times to understand it. Now, write C++ program to play text version of it! Here are a few details:

Global 2-dimensional array
```
char table[6][7];
```
can be used to represent the configuration. Each position can be space (to indicate empty), R(ed) or B(lue).

While column # is 0 to 6 internally, ask the user to input a value 1 to 7. User always plays Red, and the computer plays Blue. Both the user and the computer take turns to place pieces. Whomever gets 4 in a line first wins!

Computer need not be very intelligent - you can make it to drop a ball in random column. You can use
`rand() % 6`
to decide the column. If selected column is already full, reselect another column. If all columns are full, declare a draw!

You need to implement the following functions to complete this program:

- computerTurn() - computer randomly selects a column to put BLUE piece in. If the selected column is full, re-select another column.
- userTurn() - ask the user to select a column to put RED piece in. If the selected column is full, ask the user to select another column.
- checkConfiguration() - returns a code to indicate continue, declares draw, user wins, or computer wins!
- displayConfig() - display the contents

```
#define CONTINUE 0
#define DRAW 1
#define USER_WINS 2
#define COMPUTER_WINS 3

main()
    initialize();
    displayConfig();
    do {
        userTurn();
        displayConfig();
        if (checkConfiguration() != CONTINUE)
            break;
        computerTurn();
        displayConfig();
    } while (checkConfiguration() == CONTINUE);
    output the result;
```

Here is the sample run:

```
   1 2 3 4 5 6 7
6:
5:
4:
3:
2:
1:

User's turn: 2

   1 2 3 4 5 6 7
6:
```

```
5:
4:
3:
2:
1:   R

Computer's selection: 3

   1 2 3 4 5 6 7
6:
5:
4:
3:
2:
1:   R B

User's turn: 2

   1 2 3 4 5 6 7
6:
5:
4:
3:
2:   R
1:   R

Computer's selection: 6

   1 2 3 4 5 6 7
6:
5:
4:
3:
2:   R
1:   R B     B

so on...
```

Please submit at least 2 sample runs alongwith your program. Have fun!

Assignment 5: C++ program to do grocery checkout.

Write a C++ program to perform grocery check-out procedure for a simple store with max 100 products. Use parallel arrays to store this information.

When the program starts, it should read the product information file (inventory.txt - store the following data in a file named inventory.txt) - it contains product information (PLU code, product name, product sales type, price per pound or price per unit & current inventory level) - one product in each line.

```
4101 BRAEBURN_REG      1 0.99 101.5
4021 DELICIOUS_GDN_REG 1 0.89 94.2
4020 DELICIOUS_GLDN_LG 1 1.09 84.2
4015 DELICIOUS_RED_REG 1 1.19 75.3
4016 DELICIOUS_RED_LG  1 1.29 45.6
4167 DELICIOUS_RED_SM  1 0.89 35.4
4124 EMPIRE 1 1.14 145.2
4129 FUJI_REG 1 1.05 154.5
4131 FUJI_X-LGE 1 1.25 164.1
4135 GALA_LGE 1 1.35 187.7
4133 GALA_REG 1 1.45 145.2
4139 GRANNY_SMITH_REG 1 1.39 198.2
4017 GRANNY_SMITH_LGE 1 1.49 176.5
3115 PEACHES 1 2.09 145.5
4011 BANANAS 1 0.49 123.2
4383 MINNEOLAS 1 0.79 187.3
3144 TANGERINES 1 1.19 135.5
4028 STRAWBERRIES_PINT 0 0.99 104
4252 STRAWBERRIES_HALF_CASE 0 3.99 53
4249 STRAWBERRIES_FULL_CASE 0 7.49 67
4011 BANANAS 1 0.69 156.7
94011 ORGANIC_BANANAS 1 0.99 56.3
```

Then, program should repeatedly invoke customer check-out functionality until the store associate (user) decides to quit. As part of checkout functionality, prompt for PLU code, validate it, then the user to input weight for each product if it is sold by weight, or # of units if sold by unit. Compute the price of the item and keep up the subtotal.

Once all purchased products are rung for a customer, output the total purchase amount. If the total purchase exceeds $50, apply 5% discount to the total.

We need to keep track of inventories automatically as well. So, keep updating the inventory data along with checkout operations. When the store closes every day, product information file should be saved in output.txt in the same format as input inventory file.

CS dept attendance policy

Three consecutive absences leads to one letter grade drop. Four consecutive absences leads to an F.

UT Dallas Syllabus Policies and Procedures
The information contained in the following link constitutes the University's policies and procedures segment of the course syllabus. **Please go to**http://go.utdallas.edu/syllabus-policies **for these policies.**

The descriptions and timelines contained in this syllabus are subject to change at the discretion of the Professor.