Software Technology Department

CCPROG1 Machine Specifications

Deadline: November 28, 2023 (Tuesday) 7:59AM via AnimoSpace

Team Spirit Personalized Shirts Inc. has commissioned you to create a program to be used by their employees to track the orders for personalized shirts that are received in the shop. Part of the information that their customers need is when their orders can be picked-up, thus the program will also compute how long it takes to process the customer's order, including when they can be picked up. Below are the information needed in relation to the business processes.

An order has the same specs, i.e., same size, same position and number of add-ons. There should be a minimum of 5 pcs per order. If minimum is not met, the employee (and therefore, the program) cannot accept the order.

The neckline of the ordered shirts can be V (V-neck) or R (round neck). The possible sizes are S (small), M (medium), L (large), and X (extra large). The pricing is as follows:

- 100/pc. is the base price
- 80/pc, if minimum order is for 300 pcs and at most 500 pcs
- 75/pc, if order exceeds 500 pcs
- + 10% on the price per shirt, if size is extra large.

Since the shop caters to personalization of the shirt. There should at least be 1 add on. Add-on can be (one or a combination of):

1.) Stitch on Logo – Can be placed at the left-side or right-side on the chest or on the left or right sleeve.
   - 1 color logo costs PhP18/pc, 2 color logo costs PhP20/pc. , 3-5 color logo costs PhP25/pc
   - 10% discount is given on the logo, if minimum of 300 pcs and at most 500 pcs of shirts are ordered
   - 25% discount is given on the logo, if order exceeds 500 pcs of shirt
2.) Patch - Costs PhP15/pc. Can be placed at the left-side or right-side on the chest or on the left or right sleeve.
3.) Pocket – Costs PhP5/pc – Can be placed at the left-side or right-side on the chest, or bottom left or bottom right of the shirt.

Make sure that the chosen positions of the add-ons do not overlap. Otherwise, ask user to change the position.

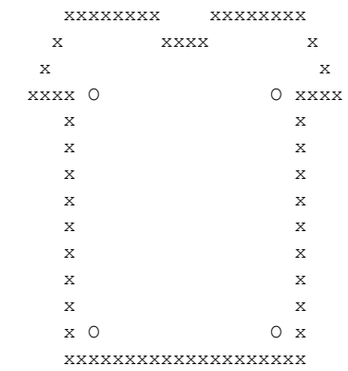
Shown below is the mock shirt showing possible places (labelled L) where the Logo can be placed. This mock shirt is size S, V-neck.

```
    xxxx    xxxx
   x    x  x    x
  x L    xx    L x
 xxxx L      L xxxx
    x          x
    x          x
    x          x
    x          x
    x          x
    xxxxxxxxxxxx
```

Shown below is the mock shirt showing possible places (labelled P) where the Patch can be placed. This mock shirt is size M, V-neck.

```
    xxxxxx    xxxxxx
   x      x  x      x
  x P      xx      P x
 xxxx P          P xxxx
    x              x
    x              x
    x              x
    x              x
    x              x
    x              x
    x              x
    xxxxxxxxxxxxxxx
```

Shown below is the mock shirt showing possible places (labelled O) where the pOcket can be placed. This mock shirt is size L, round-neck.



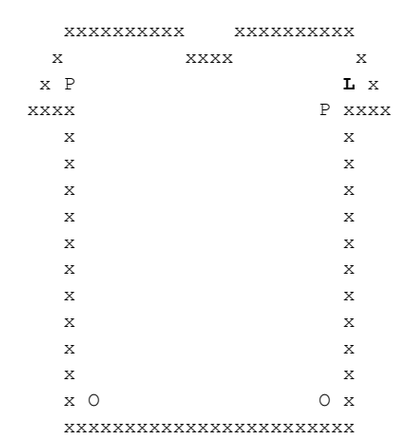
Production Schedule:

The company machine can stitch 1 logo into a shirt in 1 minute. It can stitch on 1 patch in 20 seconds. But, it takes 30 seconds to stitch on the pocket of 1 shirt. The machine is running 12 hours per day only. Assume that orders today in the shop can only be sent to the factory at the end of the business day (meaning, all orders today can start processing the next day earliest). Assume that completed shirts can be delivered back to the shop at the end of the day too (thus, pick-up from the shop can be done the next day from when the machine finished processing). So, assuming the order was made on Monday and it takes 1 hour to finish processing and it can be processed immediately on Tuesday (because orders today can be started the next day earliest), then the order is ready for pick-up on Wednesday which is **after 2 days from the order** (because the finished order will be delivered back end of Tuesday back to the store).

The program should be able to:

1.) Show options for ordering
2.) Accept information
3.) Show messages for invalid input. Assume inputs are same data types only (i.e., there is no need for program to check [and recover from] inputs that are characters when expected input is an integer).
4.) Display mock shirt using ASCII art (should also depend on size and display should be done using loops and **not brute force**). Note that display is in the perspective of holding the shirt in front of you. That is, an order where the Logo is to be placed in the Left sleeve will appear on the ASCII art as on the right, see below for example. Difference from 1 size to the next is 4 character spaces in width and 2 character spaces in length. There is no need to change the size of sleeves in the mock shirt.

    Sample below is a mock shirt where the order is for XL, round neck with 2 patches(1 at the left chest and 1 at the right sleeve) , 1 logo to be placed at the left sleeve, and 2 pockets at the bottom left and bottom right of the shirt .



5.) Show breakdown and price of each shirt, total bill, number of hours before order can be completed, and number of days before order can be picked up (considering there may be other orders). The amount paid

by the customer and the corresponding change is also computed. [Assume all orders are paid upon ordering.]

6.) Repeat Step 1-6 again if there is a next order for the day.
7.) Show end of day order summary (total number of shirts ordered and total amount received in the shop based on orders).
8.) Start new day (to reset count of orders and consider that previous day orders may have already been processed).
9.) Shutdown ordering machine for maintenance. [This is actually to exit the program.]

How to Approach the Machine Project

Step 1: Problem analysis and algorithm formulation

Read the MP Specifications again! Identify clearly what are the required information from the user, what kind of processes are needed, and what will be the output (s) of your program. Clarify with your professor any issues that you might have regarding the machine project.

When you have all the necessary information, identify the necessary functions that you will need to modularize the project. Identify the required data of these functions and what kind of data they will return to the caller. Write your algorithm for each of these modules/functions as well as the algorithm for your main program.

Step 2: Implementation

In this step, you are to translate your algorithm into proper C statements. While implementing, you are to perform the other phases of program planning and design (discussed in the other steps below) together with this step.

Follow the coding standard indicated in the course notes (Modules section in AnimoSpace).

You may choose to type your program in a text editor or an IDE (i.e. Dev-C IDE) at this point. Note that you are expected to use statements taught in class. You can explore other libraries and functions in C as long as you can clearly explain how these work. You may also use arrays, should these be applicable and you are able to properly justify and explain your implementation using these. For topics not covered, it is left to the student to read ahead, research, and explore by himself.

Note though that you are NOT ALLOWED to do the following:

- to declare and use global variables (i.e., variables declared outside any function),
- to use goto statements (i.e., to jump from code segments to code segments),
- to use the break or continue statement to exit a block. Break statement can only be used to break away from the switch block,
- to use the return statement or exit statement to prematurely terminate a loop or function or program,
- to use the exit statement to prematurely terminate a loop or to terminate the function or program, and
- to call the main() function to repeat the process instead of using loops.

It is best that you perform your coding "incrementally." This means:

- Dividing the program specification into subproblems, and solving each problem separately according to your algorithm;
- Code the solutions to the subproblems one at a time. Once you're done coding the solution for one subproblem, apply testing and debugging.

Documentation

While coding, you have to include internal documentation in your programs. You are expected to have the following:

- File comments or Introductory comments
- Function comments
- In-line comments

Introductory comments are found at the very beginning of your program before the preprocessor directives. Follow the format shown below. Note that items in between < > should be replaced with the proper information.

```
/*
     Description:      <Describe what this program does briefly>
```

```
     Programmed by:    <your name here>   <section>
     Last modified:    <date when last revision was made>
     Version:          <version number>
    [Acknowledgements: <list of sites or borrowed libraries and sources>]
*/
<Preprocessor directives>

<function implementation>

int main()
{
   return 0;
}
```

Function comments precede the function header. These are used to describe what the function does and the intentions of each parameter and what is being returned, if any. If applicable, include pre-conditions as well. Pre-conditions refer to the assumed state of the parameters. Follow the format below when writing function comments:

```
/*    <Description of function>
      Precondition: <precondition / assumption>
      @param <name> <purpose>
      @return <description of returned result>
*/
<return type>
<function name> ( <parameter list>)
      :
```

Example:

```
/* This function computes for the area of a triangle
   Precondition: base and height are non-negative values
   @param base is the base measurement of the triangle in cm
   @param height is the height measurement of the triangle in cm
   @return the resulting area of the triangle
*/
float
getAreaTri (float base,
            float height)
{
    ...
}
```

In-Line Comments are o**ther comments in major parts of the code. These are expected to explain the purpose or algorithm of groups of related code, esp. for long functions.**

STEP 3: TESTING AND DEBUGGING

SUBMIT THE LIST OF TEST CASES YOU HAVE USED. For each feature of your program, you have to fully test it before moving to the next feature. Sample questions that you should ask yourself are:

1. What should be displayed on the screen if the user inputs an order?
2. What would happen if I input incorrect inputs? (e.g., values not within the range)
3. Is my program displaying the correct output?
4. Is my program following the correct sequence of events (correct program flow)?
5. Is my program terminating (ending/exiting) correctly? Does it exit when I press the command to `quit`? Does it exit when the program's goal has been met? Is there an infinite loop?
7. and others…

IMPORTANT POINTS TO REMEMBER:

1. You are required to implement the project using the C language (C99 and NOT C++). Make sure you know how to compile and run in both the IDE (DEV-C++) and the command prompt (via
   `gcc –Wall -std=c99 <yourMP.c> -o <yourExe.exe>`
2. The implementation will require you to:

- Create and Use Functions
  Note: Non-use of self-defined functions will merit a grade of **0** for the **machine project**. Too few self-defined functions may merit deductions. A general rule is to create a separate function for each option described above, unless some features are too similar that one function can serve the purpose for two [or more] of the options. Note that functions whose tasks are only to display are not included in the count for creating user-defined functions.
- Appropriately use conditional statements, loops and other constructs discussed in class (Do not use brute force solution. **You are not allowed to use goto label statements, exit statements. You are required to pass parameters to functions and not allowed to declare global or static variables.**) Refer to Step 2 on Implementation for other details and restrictions.
- Consistently employ coding conventions
- Include internal documentation (i.e., comments)

3. Deadline for the project is the **7:59AM of November 28, 2023 (Tuesday)** via submission through **AnimoSpace**. After this time, submission facility is locked and thus no MP will be accepted anymore and this will result to a **0.0** for your machine project.
4. The following are the deliverables:

Checklist:

- ❒ Upload in AnimoSpace by clicking **Submit Assignment** on Machine Project and adding the following files:
  - ❒ source code*
  - ❒ test script**
- ❒ email the softcopies of everything as attachments to **YOUR own email address** on or before the deadline

Legend:

***Source Code also includes the internal documentation. The **first few lines of the source code** should have the following declaration (in comment) **BEFORE** the introductory comment:

```
/*****************************************************************************
This is to certify that this project is my own work, based on my personal efforts in studying and
applying the concepts learned. I have constructed the functions and their respective algorithms
and corresponding code by myself. The program was run, tested, and debugged by my own
efforts. I further certify that I have not copied in part or whole or otherwise plagiarized the
work of other students and/or persons.
                                        <your full name>, DLSU ID# <number>
*****************************************************************************/
```

****Test Script should be in a table format, with header as shown below. There should be **at least 3 distinct test classes** (as indicated in the description) **per function**. There is no need to create test scripts for functions that only perform displaying on screen (like menu where there are no conditions being checked; but for the ASCII art which is displayed depending on size and on add-ons, there are conditions checked, thus should be included in the test script).

Function Name	#	Test Description	Sample Input (either from the user or to the function)	Expected Result	Actual Result	P/F
getAreaTri	1	**base and height measurements are less than 1.**	**base = 0.25** **height = 0.75**	**…**	**…**	
	2	:::				
	3					

Test descriptions are supposed to be unique and should indicate classes/groups of test cases on what is being tested. Given the function getAreaTri(), the following are 3 distinct classes of tests:

i.) testing with base and height values smaller than 1
ii.) testing with whole number values for base and height
iii.) testing with floating point number values for base and height, larger than 1.

The following test descriptions are incorrectly formed:

Too specific: testing with base containing 0.25 and height containing 0.75
Too general: testing if function can generate correct area of triangle
Not necessary -- since already defined in pre-condition: testing with base or height containing negative values

5. **MP Demo:** You will demonstrate your project on a specified schedule during the last weeks of classes. Being unable to show up on time during the demo or being unable to answer convincingly the questions during the demo will merit a grade of **0.0** for the **MP**. The project is initially evaluated via black box testing (i.e., based on output of running program). Thus, if the program does not compile successfully using gcc -Wall -std=c99 and execute in the command prompt, a grade of 0 for the project will be incurred. However, a fully working project does not ensure a perfect grade, as the implementation (i.e., correctness and compliance in code) is still checked.
6. Any requirement not fully implemented and instruction not followed will merit deductions.
7. This is an **individual project**. Working in collaboration, asking other people's help, and/or copying other people's work are considered as cheating. Cheating is punishable by a grade of **0.0** for CCPROG1 course, aside from which, a cheating case may be filed with the Discipline Office.
8. The above description of the program is the basic requirement. A maximum of 10 points will be given as bonus. Use of colors may not necessarily incur bonus points. Sample additional features could be:
   (a) properly and substantially using arrays in the implementation of the project (3 pts),
   (b) program having save AND load of order information to files (4 pts),
   (c) tracking each order information and presenting the set of orders that are ready for pick-up at the start of a new day (3 pts). Some of the indicated additional features may require self-study.

   Note that any additional feature not stated here may be added but **should not conflict with whatever instruction was given in the project specifications**. Bonus points are given upon the discretion of the teacher, based on the difficulty and applicability of the feature to the program. Note that **bonus points can only be credited if all the basic requirements are fully met** (i.e., complete and no bugs).

HONESTY POLICY AND INTELLECTUAL PROPERTY RIGHTS

Honesty policy applies. Please take note that you are NOT allowed to borrow and/or copy-and-paste – in full or in part any existing related program code from the internet or other sources (such as printed materials like books, or source codes by other people that are not online). **You should develop your own codes from scratch by yourself.**

SAMPLE RUN:

Please find below a sample guide on the **flow** of the program. Those in bold font are user inputs. Note that for the project, you can add/edit the screen design and presentation.

```
Main Menu
1 - Order
2 - Show Order Summary for the day
3 - Start New Day
4 - Shutdown for Maintenance

Enter option: 1


Order Form
Enter size: S
Enter number of pieces: -3
Invalid number of pieces.

Enter number of pieces: 3
Too few. Invalid number of pieces.

Enter number of pieces: 50
Enter choice for neckline: P
Invalid choice.

Enter choice for neckline: r
```

Can show options for the inputs (like for the neckline and sizes).

```
Add-on options:
L – Logo
P – Patch
O – pOcket

Enter add-on: t
Invalid choice.

Enter add-on: p

Patch placement options:
1 – left chest
2 – right chest
3 – left sleeve
4 – right sleeve

Enter patch placement: 1

Add another patch? Y

Patch placement options:
1 – left chest
2 – right chest
3 – left sleeve
4 – right sleeve

Enter patch placement: 4

Add another patch? N

Add another add-on? Y

Add-on options:
L – Logo
P – Patch
O – pOcket

Enter add-on: o

Pocket placement options:
1 – left chest
2 – right chest
3 – bottom left
4 – bottom right

Enter pocket placement: 1
Invalid. Already indicated to place a patch.

Enter pocket placement: 3

Add another pocket? N

Add another add-on? Y

Add-on options:
L – Logo
P – Patch
O – pOcket

Enter add-on: o

Pocket placement options:
1 – left chest
2 – right chest
3 – bottom left
4 – bottom right
```

Input sequence for the order may be different (i.e., number of pieces can be last OR even allow the user to determine which input to give first, via a menu. Refer to the following for an example)

Order Options:

1- Enter quantity
2- Enter neckline choice
3- Enter add-ons
4- Enter size

Enter option:

Unless specified in the text description of the MP specs, you can decide (or modify) the input needed. For example, since it is not indicated in the specs on the indicated code to represent the option for the placement of the add-ons, you can decide if you want to have a number or a character to represent the location. That is, for example, you could choose to say 'A' is for left chest, 'B' is for right chest, …

```
Enter pocket placement: 3
Invalid. Already indicated to place a pocket.

Enter pocket placement: 4

Add another pocket? N

Add another add-on? Y

Add-on options:
L – Logo
P – Patch
O – pOcket

Enter add-on: L

Logo placement options:
1 – left chest
2 – right chest
3 – left sleeve
4 – right sleeve

Enter logo placement: 3
How many colors is the logo: 1

Add another logo? N

Add another add-on? N

Order Summary:
50 pcs of Small Round-neck shirt with        P    100.00 @
   Logo on left sleeve, 1 color              P     18.00 @
   Patch on left chest                       P     15.00 @
   Patch on right sleeve                     P     15.00 @
   Pocket on bottom left of shirt            P      5.00 @
   Pocket on bottom right of shirt           P      5.00 @
-------------------------------------------------------------------
                                             P    158.00 @  x  50
                     Total Bill for Order    P   7900.00

Mock Shirt:
    xxxx    xxxx
   x    xxxx    x
  x P         L x
 xxxx        P xxxx
    x          x
    x          x
    x          x
    x          x
    x O      O x
    xxxxxxxxxxx

Production Time: 2 hours 13 minutes 20 seconds
Order can be picked up after 2 days

Confirm order? Y

Amount paid by customer: 8000.00

Total change: P 100.00
Breakdown of change:
  Php100 – 1

Main Menu
1 – Order
2 - Show Order Summary for the day
3 – Start New Day
4 – Shutdown for Maintenance

Enter option: 1
```

You may choose how you design your screen display and format, but make sure that data are easily seen. Employ formatting and white spaces for readability.

Notice that those that result to 0 count for the denomination of money is not supposed to be displayed.

```
Order Form
Enter size: M
Enter number of pieces: 555
Enter choice for neckline: v

Add-on options:
L - Logo
P - Patch
O - pOcket

Enter add-on: l

Logo placement options:
1 - left chest
2 - right chest
3 - left sleeve
4 - right sleeve

Enter logo placement: 1
How many colors is the logo: 2

Add another logo? Y

Logo placement options:
1 - left chest
2 - right chest
3 - left sleeve
4 - right sleeve

Enter logo placement: 3
How many colors is the logo: 3

Add another add-on? N

Order Summary:
555 pcs of Medium V-neck shirt with          P     75.00 @
   Logo on left chest, 2 color               P     15.00 @
   Logo on left sleeve, 3 color              P     18.75 @
-------------------------------------------------------------------------
                                             P    108.75 @  x  555
                     Total Bill for Order    P  60356.25

Mock Shirt:
    xxxxxx   xxxxxx
   x     x  x     x
  x       xx     L x
 xxxx           L xxxx
    x            x
    x            x
    x            x
    x            x
    x            x
    x            x
    x            x
    xxxxxxxxxxxxxxx

Production Time: 18 hours 30 minutes
Order can be picked up after 3 days

Confirm order? Y

Amount paid by customer: 61000.00

Total change: P 643.75
Breakdown of change:
  Php500 - 1
  Php100 - 1
  Php20  - 2
  Php1   - 3
```

```
  Php0.50- 1
  Php0.25- 1

Main Menu
1 - Order
2 - Show Order Summary for the day
3 - Start New Day
4 - Shutdown for Maintenance

Enter option: 2

Order Summary for the day:
Total number of shirts ordered: 605
Total amount received: P68256.25

Main Menu
1 - Order
2 - Show Order Summary for the day
3 - Start New Day
4 - Shutdown for Maintenance

Enter option: 3

Starting new day…

Main Menu
1 - Order
2 - Show Order Summary for the day
3 - Start New Day
4 - Shutdown for Maintenance

Enter option: 2

Order Summary for the day:
Total number of shirts ordered: 0
Total amount received: P0.00

Main Menu
1 - Order
2 - Show Order Summary for the day
3 - Start New Day
4 - Shutdown for Maintenance

Enter option: 4

Program terminating for maintenance…
```