XIAMEN UNIVERSITY MALAYSIA

Course Code	:	CST209
Course Name	:	Object-Oriented Programming – C++
Lecturer	:	Teo Bee Guan
Academic Session	:	2023/04
Assessment Title	:	Final Project
Submission Due Date	:	July 16th, 2023

Prepared by :	Student ID	Student Name
	AIT2109821	Krisnadi Michael

Date Received   : July 16th, 2023

Feedback from Lecturer:

Mark:

Own Work Declaration

I/We hereby understand my/our work would be checked for plagiarism or other misconduct, and the softcopy would be saved for future comparison(s).

I/We hereby confirm that all the references or sources of citations have been correctly listed or presented and I/we clearly understand the serious consequence caused by any intentional or unintentional misconduct.
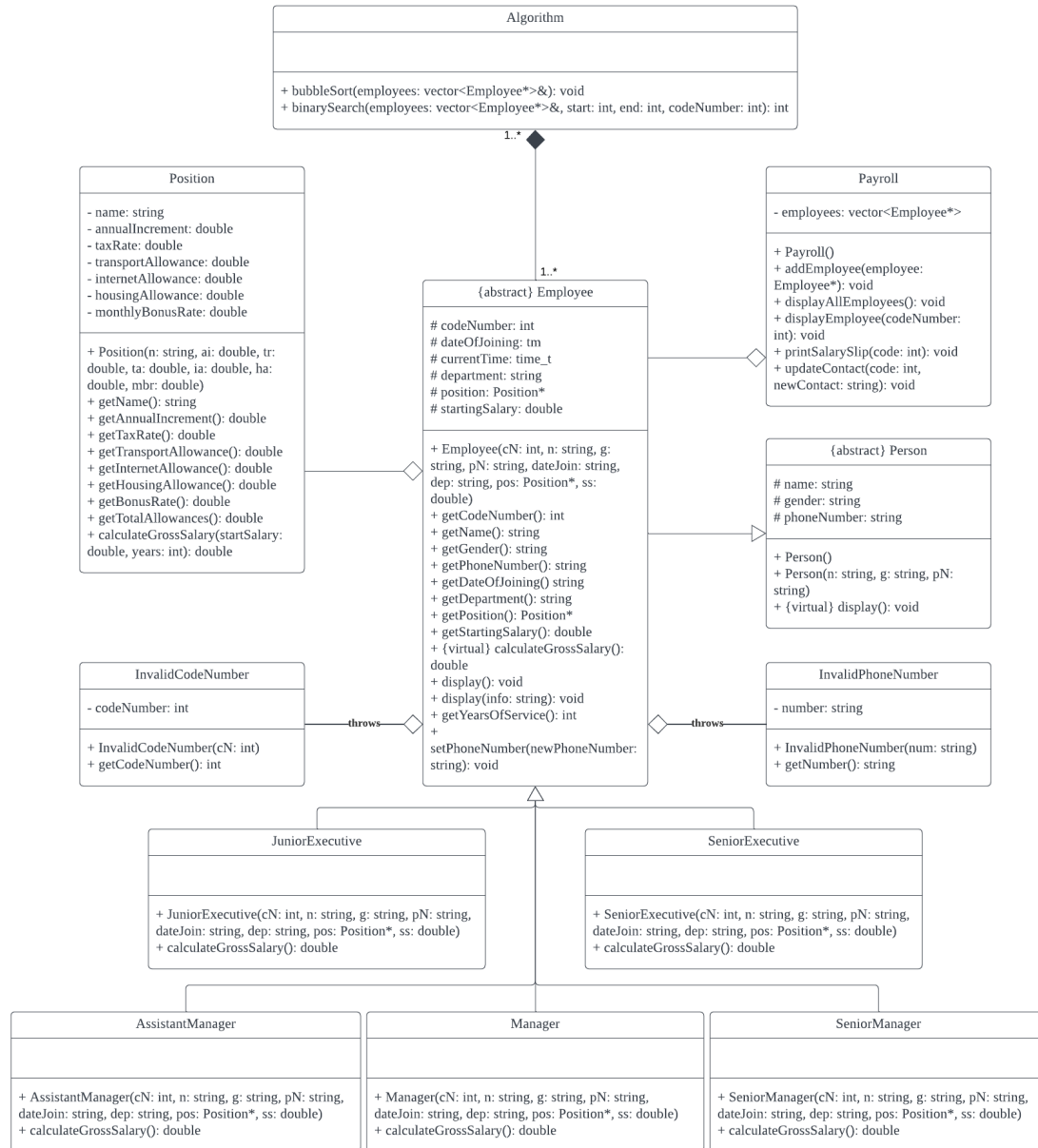
This work is not made on any work of other students (past or present), and it has not been submitted to any other courses or institutions before.
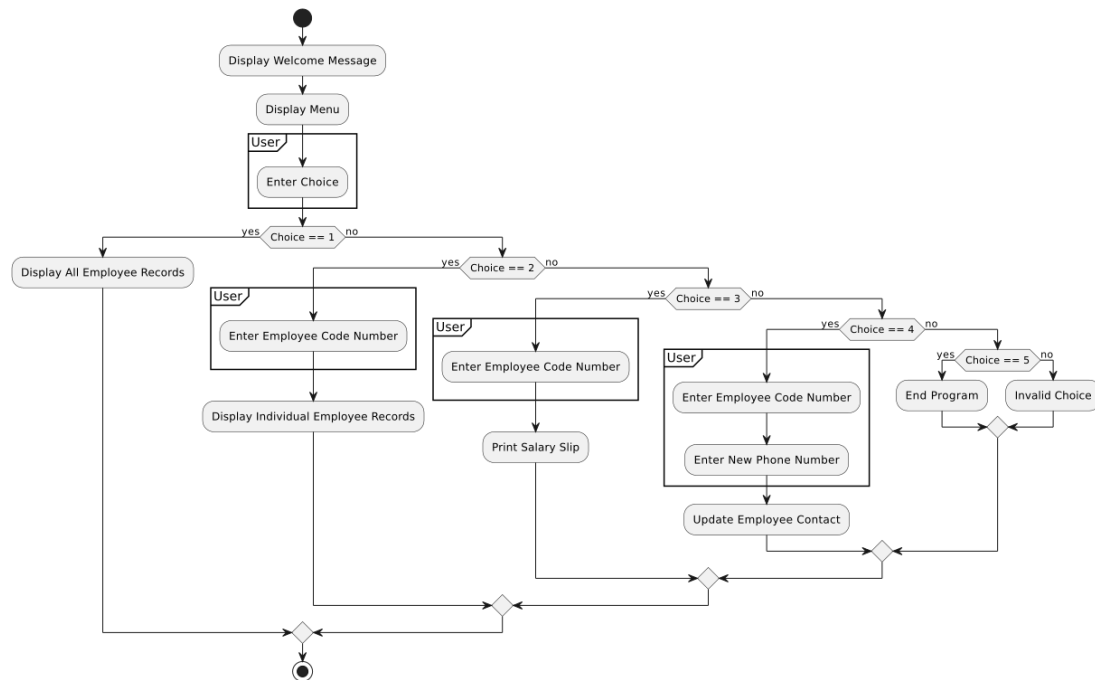
Signature:

Date: July 16th, 2023

UML Class Diagram

Algorithm

+ bubbleSort(employees: vector<Employee*>&): void
+ binarySearch(employees: vector<Employee*>&, start: int, end: int, codeNumber: int): int

1..*

Position

- name: string
- annualIncrement: double
- taxRate: double
- transportAllowance: double
- internetAllowance: double
- housingAllowance: double
- monthlyBonusRate: double

+ Position(n: string, ai: double, tr: double, ta: double, ia: double, ha: double, mbr: double)
+ getName(): string
+ getAnnualIncrement(): double
+ getTaxRate(): double
+ getTransportAllowance(): double
+ getInternetAllowance(): double
+ getHousingAllowance(): double
+ getBonusRate(): double
+ getTotalAllowances(): double
+ calculateGrossSalary(startSalary: double, years: int): double

1..*

{abstract} Employee

codeNumber: int
dateOfJoining: tm
currentTime: time_t
department: string
position: Position*
startingSalary: double

+ Employee(cN: int, n: string, g: string, pN: string, dateJoin: string, dep: string, pos: Position*, ss: double)
+ getCodeNumber(): int
+ getName(): string
+ getGender(): string
+ getPhoneNumber(): string
+ getDateOfJoining() string
+ getDepartment(): string
+ getPosition(): Position*
+ getStartingSalary(): double
+ {virtual} calculateGrossSalary(): double
+ display(): void
+ display(info: string): void
+ getYearsOfService(): int
+ setPhoneNumber(newPhoneNumber: string): void

Payroll

- employees: vector<Employee*>

+ Payroll()
+ addEmployee(employee: Employee*): void
+ displayAllEmployees(): void
+ displayEmployee(codeNumber: int): void
+ printSalarySlip(code: int): void
+ updateContact(code: int, newContact: string): void

{abstract} Person

name: string
gender: string
phoneNumber: string

+ Person()
+ Person(n: string, g: string, pN: string)
+ {virtual} display(): void

InvalidCodeNumber

- codeNumber: int

+ InvalidCodeNumber(cN: int)
+ getCodeNumber(): int

throws

InvalidPhoneNumber

- number: string

+ InvalidPhoneNumber(num: string)
+ getNumber(): string

throws

JuniorExecutive

+ JuniorExecutive(cN: int, n: string, g: string, pN: string, dateJoin: string, dep: string, pos: Position*, ss: double)
+ calculateGrossSalary(): double

SeniorExecutive

+ SeniorExecutive(cN: int, n: string, g: string, pN: string, dateJoin: string, dep: string, pos: Position*, ss: double)
+ calculateGrossSalary(): double

AssistantManager

+ AssistantManager(cN: int, n: string, g: string, pN: string, dateJoin: string, dep: string, pos: Position*, ss: double)
+ calculateGrossSalary(): double

Manager

+ Manager(cN: int, n: string, g: string, pN: string, dateJoin: string, dep: string, pos: Position*, ss: double)
+ calculateGrossSalary(): double

SeniorManager

+ SeniorManager(cN: int, n: string, g: string, pN: string, dateJoin: string, dep: string, pos: Position*, ss: double)
+ calculateGrossSalary(): double

Activity Flow Diagram

1. When the program starts, it displays a welcome message to the user "Welcome to Payroll Management System"
2. At the same time, the main menu is also displayed to the user. It consists of five options for the users to choose from.
3. The user is prompted to enter a number in order to do the operation they want to do
4. The program checks the user's choice:
   a. If the user's input is 1, then the program will display the records of all the employee
   b. If the user's input is 2, then the program will display an individual employee record. Before that, it will ask the user to enter the employee's code number that the user wants the program to display.
   c. If the user's input is 3, the program will ask the user to enter one of the employees' code numbers and will print the corresponding employee's salary slip
   d. If the user's input is 4, the program will ask the user to enter one of the employees' code numbers and then ask the user again to enter a new phone number that is going to be assigned to the corresponding employee. Then it will update that employee's contact, that is, their phone number.
   e. If the user's input is 5, the program will terminate
   f. If the user's input is invalid, that is, any input other than the number 1 through 5, the program will show a message saying "Invalid Choice".

MARKING RUBRICS

Component Title	Building a Payroll Management Program				Percentage (%)	80	
Criteria	**Score and Descriptors**					Weight (%)	Marks
	Excellent (9-10)	Good (7-8)	Average (5-6)	Need Improvement (3-4)	Poor (0-2)		
Display all employee records	Accurate use of variables and design of function within the class	Generally proper use of variables and design of function with only few minor mistakes	Acceptable usage of variables and design of function but with few noticeable major mistakes	The function does not work as intended but some logic still partially correct	The function does not work as intended and the logic is almost inaccurate or no attempt at all.	10	
	Excellent (13-15)	Good (10-12)	Average (7-9)	Need Improvement (4-6)	Poor (0-3)		
Display individual employee record.	Accurate use of variables and design of function within the class	Generally proper use of variables and design of function with only few minor mistakes	Acceptable usage of variables and design of function but with few noticeable major mistakes	The function does not work as intended and some logic still partially correct	The function does not work as intended and the logic is almost inaccurate or no attempt at all.	15	
	Excellent (25-30)	Good (19-24)	Average (13-18)	Need Improvement (7-12)	Poor (0-6)		
Print employee salary slip	Accurate use of variables and design of function within the class	Generally proper use of variables and design of function with only few minor mistakes	Acceptable usage of variables and design of function but with few noticeable major mistakes	The function does not work as intended but some logic still partially correct	The function does not work as intended and the logic is almost inaccurate or no attempt at all.	30	
	Excellent (9-10)	Good (7-8)	Average (5-6)	Need Improvement (3-4)	Poor (0-2)		
Update Employee Contact	Accurate use of variables and design of main function	Generally proper use of variables and design of main function with only few minor mistakes	Acceptable usage of variables and design of main function but with few noticeable major mistakes	The main function does not work as intended but some logic still partially correct	The main function does not work as intended and the logic is almost inaccurate.	10	
	Excellent (13-15)	Good (10-12)	Average (7-9)	Need Improvement (4-6)	Poor (0-3)		
Main Screen	Accurate use of variables and design of function within the class	Generally proper use of variables and design of function with only few minor mistakes	Acceptable usage of variables and design of function but with few noticeable major mistakes	The function does not work as intended but some logic still partially correct	The function does not work as intended and the logic is almost inaccurate or no attempt at all.	15	
					TOTAL	80	

Component Title	Project Documentation					Percentage (%)	20
Criteria	Score and Descriptors					Weight (%)	Marks
	Excellent (9-10)	Good (7-8)	Average (5-6)	Need Improvement (3-4)	Poor (0-2)		
UML Diagram	Clear and accurate usage of diagram	Few minor mistakes found in the diagram (e.g. wrong class name).	Some noticeable mistakes (e.g. inaccurate class relationship)	Not all the classes are included in the diagram.	The diagram is confusing, and the format is not standardized. Or no attempt at all.	10	
	Excellent (9-10)	Good (7-8)	Average (5-6)	Need Improvement (3-4)	Poor (0-2)		
Activity Flow Diagram	Clear and accurate usage of diagram	A few minor mistakes found in the diagram (e.g. wrong class/function name).	Some noticeable mistakes (e.g.logical flow)	Some important logical flows are not included in diagram.	The diagram is confusing, and the format is not standardized. Or no attempt at all.	10	
					TOTAL	20	