סדנא ב- C++ – 150018

תרגיל בית מספר 6

ירושה

<u>שימו לב:</u>

א. הקפד/י על קְרִיאוּת התכנית ועל עימוד (Indentation).
ב. הקפד/י לבצע בדיוק את הנדרש בכל שאלה.
ג. בכל אחת מהשאלות יש להגדיר פונקציות במידת הצורך עבור קריאות התכנית.
ד. יש להגיש את התרגיל על פי ההנחיות להגשת תרגילים (המופיע באתר הקורס) וביניהם:
השתמש/י בשמות משמעותיים עבור המשתנים.
יש לתעד את התכנית גם עבור פונקציות אותם הנך מגדיר/ה וכן על תנאים ולולאות וקטעי קוד מורכבים, ובנוסף, **דוגמת הרצה לכל תכנית בסוף הקובץ**!
הגשה יחידנית - אין להגיש בזוגות.

הערה חשובה: לכל תרגיל בית מוגדר <u>שבוע אחד בלבד</u> להגשה, אלא אם כן קיבלת הוראה אחרת מהמרצה שלך.
<u>תיבות ההגשה הפתוחות לא מהוות היתר להגשה באיחור.</u>

<u>שאלה מס' 1:</u>

במכללת "נלמד בכיף", קיימים שני סוגי עובדים:

- עובד במשרה מלאה (fulltime)
- ועובד במשרה חלקית (parttime)

עבור כל עובד רוצים לשמור את הנתונים הבאים:

- שם פרטי של העובד
- ת.ז של העובד
- מספר שנות הוותק של העובד בעבודה
- אחוז ההפרשה למס הכנסה

מדיניות התשלום במכללה היא פעם בחודש, לפי הנתונים הבאים:

- עבור עובד במשרה מלאה, החישוב הוא משכורתו השנתית לחלק ל-12.
  - לכן עבור עובד במשרה מלאה יש לשמור גם את משכורתו השנתית
- עבור עובד במשרה חלקית, החישוב הוא שעות העבודה בחודש כפול התשלום שהובטח לו לשעת עבודה.
  - לכן עבור עובד במשרה חלקית יש לשמור גם שעות עבודה ותשלום לשעה.

הנהלת המכללה החליטה לתת לכל עובד בונוס לחודש תשרי, לפי הנוסחה הבאה:

- במידה והעובד הוא בעל וותק בעבודה של 5 שנים או פחות, הבונוס יהיה 500 שקל.
- במידה והעובד בעל וותק בעבודה של יותר מ- 5 שנים, הבונוס יהיה 25% ממשכורתו החודשית לפני הבונוס.

לאור הנתונים לעיל:

א. הגדר/י מחלקה בסיסית בשם Employee, המייצגת עובד במכללה.
המחלקה תכלול את התכונות הבאות:

- name – שמו הפרטי של העובד (string)
- id – ת.ז של העובד (int)
- seniority – שנות וותק של העובד (int)
- pay – שכר לחודש (float)

וכן את המתודות/ הבנאים הבאים:

- constructor עבור אתחול כל תכונות המחלקה. שם (name), ת.ז. (id) ושנות וותק (seniority) יאותחלו לפי הערכים שיתקבל כפרמטרים ואת התכונה משכורת (pay)יש לאתחל ב-0).
- empty constructor – (בנאי ריק) עם ערכי ברירת מחדל, מחרוזת ריקה ואפסים בהתאמה.
- copy constructor (בנאי העתקה).
get ו- set עבור הצבה ואחזור של תכונות המחלקה – במידת הצורך.
- salaryAfterBonus – לעדכון המשכורת עבור בונוס תשרי. (שימו לב: המחלקות היורשות מהמחלקה Employee **לא ידרסו** את המתודה הזו).
- אופרטור << עבור קליטת נתוני העובד (יש לקלוט את התכונות עפ"י סדר הגדרתן במחלקה) בפורמט הבא:

  Enter employee details:

- אופרטור >> עבור הדפסת נתוני העובד בפורמט הבא:

  Employee:
  Employee ID:
  Years Seniority:

ב. הגדר/י מחלקה בשם FullTime המייצגת עובד במשרה מלאה, היורשת ממחלקת Employee.
המחלקה תכלול (בנוסף לתכונות של Employee) את התכונה הבאה:

- salaryPerYear – משכורת השנתית של העובד (float),

וכן את המתודות/ הבנאים הבאים:

- constructor – המקבל כפרמטרים את פרטי העובד כולל משכורת שנתית (שימו לב, תכונה נוספת למחלקה זו) ומאתחל את התכונות עם ערכים אלו. הבנאי יחשב ערך עבור המשתנה pay (שנוצר במחלקת Employee ושומרת תשלום חודשי לעובד) על ידי קריאה למתודה salary (פרטים של המתודה בהמשך).

- empty constructor – (בנאי ריק) עם ערכי ברירת מחדל, מחרוזת ריקה ואפסים בהתאמה.
- copy constructor (בנאי העתקה)
- get ו- set עבור הצבה ואחזור של תכונות המחלקה – במידת הצורך.
- salary – לחישוב משכורת חודשית של העובד.
- *שאלה למחשבה – לא מגדירים כאן את המתודה של salaryAfterBonus. מה יקרה בזמן ריצה של התוכנית?*
- אופרטור << עבור קליטת נתוני העובד (יש לקלוט את התכונות עפ"י סדר הגדרתן במחלקה)

Enter employee details:

שימו לב: לא קולטים ערך עבור המשתנה pay (שנוצר במחלקת Employee ושומר תשלום חודשי לעובד), אלא, אחרי קריאת כל שאר הנתונים של העובד יש לקרוא למתודה salary כדי להציב במשתנה pay ערך .

- אופרטור >> עבור הדפסת נתוני העובד (שם, ת.ז., שנות וותק, ומשכורת **לחודש**) בפורמט הבא:

Employee:
Employee ID:
Years Seniority:
Salary per Month:

<u>שימו לב! יש להימנע משכפול קוד ואין לכתוב במחלקה זו קוד שאינו נצרך. כלומר, במידה וקיימת מתודה או חלק מהקוד שבה אצל האב – אין לכתוב קטע קוד נוסף המבצע את אותה פעולה אצל הבן!!</u>

ג. באופן דומה, הגדירו מחלקה בשם PartTime המייצגת עובד במשרה חלקית, היורשת ממחלקת Employee.
המחלקה תכלול (בנוסף לתכונות של Employee) את התכונות הבאות:

- hoursOfWork – מספר שעות שהעובד עבד בחודש (int)
- payPerHour – משכורת לשעה של העובד (float)

וכן את המתודות/ הבנאים הבאים:

- constructor – המקבל כפרמטרים את פרטי העובד כולל שעות עבודה ותשלום לשעה (שימו לב, תכונות נוספות למחלקה זו) ומאתחל את התכונות עם ערכים אלו. הבנאי יחשב ערך עבור המשתנה pay (שנוצר במחלקת

Employee ושומר תשלום חודשי לעובד) על ידי קריאה למתודה salary (פרטים על המתודה בהמשך).

- empty constructor (בנאי ריק) – עם ערכי ברירת מחדל, מחרוזת ריקה ואפסים בהתאמה.
- copy constructor (בנאי העתקה).
- get ו- set עבור הצבה ואחזור של תכונות המחלקה – במידת הצורך.
- salary – לחישוב משכורת חודשית של העובד.
- *שאלה למחשבה – לא מגדירים כאן את המתודה salaryAfterBonus. מה יקרה בזמן ריצה של התוכנית?*
- אופרטור << עבור קליטת נתוני העובד (יש לקלוט את התכונות עפ"י סדר הגדרתן במחלקה)

  Enter employee details:

שימו לב: לא קולטים ערך עבור המשתנה pay (שנוצר במחלקת Employee ושומר תשלום חודשי לעובד), אלא, אחרי קריאת כל הנתונים האחרים של העובד יש לקרוא למתודה salary כדי להציב במשתנה pay ערך .

- אופרטור >> עבור הדפסת נתוני העובד העובד (שם, ת.ז., שנות וותק, שעות שעבד בחודש ומשכורת **לחודש**) בפורמט הבא:

  Employee:
  Employee ID:
  Years Seniority:
  Hours:
  Salary per Month:

<u>שימו לב! יש להימנע משכפול קוד ואין לכתוב במחלקה זו קוד שאינו נצרך. כלומר, במידה וקיימת מתודה או חלק מהקוד שבה אצל האב – אין לכתוב קטע קוד נוסף המבצע את אותה פעולה אצל הבן!!</u>

בכל מתודה בה עלולה להופיע שגיאה יש לזרוק exception ERROR.
שימו לב, במקרה של שגיאה, למרות השגיאה, יש לקלוט את כל נתוני העובד כלומר קודם קולטים את כל הנתונים ואחר כך בודקים וזורקים חריגות.

נתונה התכנית הראשית הבאה, הבוחנת את נכונות המחלקה:

```
#include "FullTime.h"
#include "PartTime.h"
#include <iostream>
using namespace std;
int main()
{
	FullTime arrF[3];
	for (int i = 0; i < 3; i++)
	{
		try
		{
			cin >> arrF[i];
		}
		catch (const char* str)
		{
			cout << str << endl;
			i--;
		}
	}


	PartTime arrP[3];
	for (int i = 0; i < 3; i++)
	{
		try
		{
			cin >> arrP[i];
		}
		catch (const char* str)
		{
			cout << str << endl;
			i--;
		}
	}


	for (int i = 0; i < 3; i++)
	{
		cout << arrF[i];
		cout << "After Bonus: " << arrF[i].salaryAfterBonus() << endl;
	}


	for (int i = 0; i < 3; i++)
	{
		cout << arrP[i];
		cout << "After Bonus: " << arrP[i].salaryAfterBonus() << endl;
	}
	return 0;
}
```

דוגמא להרצת התכנית עם 3 עובדים (שניים הראשונים סוג FullTime והשלישי מסוג PartTime):

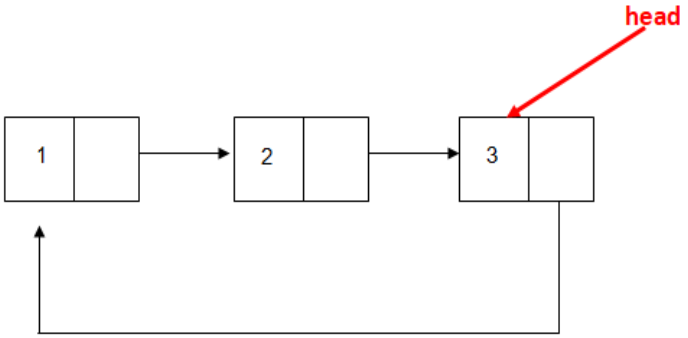
```
Enter employee details:
moshe  1234 3 36000
Enter employee details:
miriam  4321 8 48000
Enter employee details:
aharon  5678 4 160 35

Employee: moshe
Employee ID: 1234
Years Seniority: 3
Salary per Month: 3000
After Bonus: 3500

Employee: miriam
Employee ID: 4321
Years Seniority: 8
Salary per Month: 4000
After Bonus: 5000

Employee: aharon
Employee ID: 5678
Years Seniority: 4
Hours: 160
Salary per Month: 5600
```

<u>שאלה מס' 2:</u>

רשימה מעגלית RoundList, הינה רשימה לינארית שבה החוליה האחרונה מצביעה על החוליה הראשונה.
לדוגמה: נניח שברשימה RoundList יש שלושה איברים -

- 1 הוא איבר הראשון
- 2 הוא איבר השני
- 3 היא איבר השלישי

אז הרשימה המקושרת היתה נראית כך:



כלומר האיבר הראשון מצביע לאיבר השני, האיבר השני מצביע לאיבר השלישי, האיבר השלישי (האיבר האחרון) מצביע לאיבר הראשון וראש הרשימה (head) מצביע על האיבר האחרון. שימו לב בתוך הרשימה המעגלית יש אך ורק head אחד.
על רשימה מעגלית ניתן לבצע את כל המתודות המוגדרות עבור רשימה לינארית,
ובנוסף ניתן לבצע את המתודות הבאות:

- הוספת איבר בסוף הרשימה addToEnd(int val). המתודה מקבלת כפרמטר מספר שלם val, ומוסיפה חוליה בסוף הרשימה עם הערך val.
- חיפוש ברשימה search(int n). המתודה מקבלת כפרמטר מספר שלם לא שלילי, n. המתודה מחזירה את ערכו של האיבר הנמצא במקום n ברשימה המעגלית. (שימו לב: n יכול להיות מספר גדול יותר ממספר האיברים הקיימים ברשימה. מאחר והרשימה מעגלית יש להמשיך למנות את האיברים בשנית עד לאינדקס n. מספור הרשימה מתחיל מאינדקס 0). במידה והרשימה ריקה, המתודה מחזירה 1-.
- operator<< עבור הדפסת איברי הרשימה המעגלית. יש להתחיל את ההפדסה באיבר הראשון של הרשימה ולהדפיס רווח בין כל איברי הרשימה. במידה שהרשימה ריקה לא יודפס כלום.

הגדר/י את המחלקה RoundList כמחלקה היורשת מ-List (המחלקה List המגדירה רשימה לינארית שנלמדה בהרצאה). ממש/י את כל המתודות הנדרשות מרשימה מקושרת וגם את שלש המתודות הנוספות עבור רשימה מקושרת מעגלית.

הנחיות נוספות:

- **אין להוסיף תכונות פרטיות חדשות למחלקה RoundList – התכונה היחידה במחלקה תהיה מצביע לראש הרשימה כפי שמוגדר במחלקה List הבסיסית.**
- תזכורת הבנאי של RoundList יקרא לבנאי של List והפונקציה ההורסת של RoundList תקרא לפונקציה הורסת של List. יש לקחת את זה בחשבון כשכותבים את הבנאים/פונקציה הורסת של RoundList.
- עליך להכריע אילו מתודות במחלקה List יש לדרוס במחלקה RoundList היורשת, ואילו מתודות אין צורך לדרוס.

נתונה התכנית הראשית הבאה, הבוחנת את נכונות המחלקה:

```
#include "RoundList.h"
#include <iostream>
using namespace std;

enum CHOICES {
	EXIT, ADD, ADD_TO_END, REMOVE_FIRST, SEARCH, CLEAR, EMPTY, PRINT
};

int main() {
	RoundList roundList;
	int choice;
	cout << "Enter your choice: \n";
	cin >> choice;
	while (choice != EXIT) {
		int num;
		switch (choice) {
		case ADD:
			cout << "Enter a number: \n";
			cin >> num;
			roundList.add(num);
			break;
		case ADD_TO_END:
			cout << "Enter a number: \n";
			cin >> num;
			roundList.addToEnd(num);
			break;
		case REMOVE_FIRST:
			roundList.removeFirst();
			break;
		case SEARCH:
			cout << "Enter a number: ";
			cin >> num;
			cout << roundList.search(num) << endl;
			break;
		case CLEAR:
			roundList.clear();
			break;
		case EMPTY:
			if (roundList.isEmpty())
				cout << "Empty" << endl;
			else
				cout << "Not empty" << endl;
			break;
		case PRINT:
			cout << roundList << endl;
			break;
		default: cout << "ERROR!" << endl;
		}
		cout << "Enter your choice: \n";
		cin >> choice;
	}
	return 0;
}
```

דוגמא להרצת התכנית: דוגמא להרצת התכנית

```
Enter your choice:
1
Enter a number:
 10
Enter your choice:
 1
Enter a number:
 11
Enter your choice:
7
11 10
Enter your choice:
2
Enter a number:
 20
Enter your choice:
2
Enter a number:
 21
Enter your choice:
7
20 21 11 10
Enter your choice:
6
Not empty
Enter your choice:
3
Enter your choice:
7
21 11 10
Enter your choice:
4
Enter a number:
1
11
Enter your choice:
5
Enter your choice:
6
Empty
Enter your choice:
```