סדנא ב- C++ – 150018

תרגיל בית מספר 11

קבצים בינאריים

<u>שים/י 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 (ולא יאוחר ממועד הבחינה).
<u>תיבות ההגשה הפתוחות לא מהוות היתר להגשה באיחור.</u>

<u>שאלה מס' 1:</u>

בתרגיל זה עלייך לבצע גישה אקראית לקובץ בינארי (Random Access Binary Files).

התכנית מאתחלת קובץ בינארי עבור 100 רשומות זהות בגודלן, ולכל אחת מהן מפתח ראשי ייחודי הממוספר מ-1 עד 100, בהתאם למיקומה בקובץ (המספור מתחיל מ-1, כלומר – לרשומה השנייה יהיה את המפתח 2 ולרשומה האחרונה את המפתח 100).
בתחילה הרשומות ריקות, ובמהלך התכנית מתבצעת גישה ישירה לרשומה, לפי דרישת המערכת (מאחר והרשומות זהות בגודלן, ניתן לדעת את מיקומה של כל רשומה בקובץ, בהינתן המפתח הראשי שלה). לאור זאת – ניתן להניח שבמידה והמפתח הראשי של רשומה הוא 0, הרשומה ריקה. ולעומת זאת אם המפתח הראשי שונה מ-0, הרשומה קיימת במערכת.

כתוב/י תכנית אשר תנהל את מערכת הרישום של ילדים בחוגים במקום מגוריהם.

1. הגדר/י מחלקה Family המתאימה לשמירת נתוניהם של כל משפחה. הנתונים הם:
   א. מס' משפחה (מס' בטווח . 100-1. מס' המהווה מפתח ייחודי)
   ב. שם משפחה (עד 20 תווים)
   ג. מס' נפשות במשפחה
   ד. מס' טלפון
   ה. מערך המכיל סימון בוליאני עבור כל אחד מ 6 החוגים הפעילים בשכונה. אמת- במידה ואחד מילדי המשפחה רשום לחוג, אחרת שקר.
2. כתוב/י פונקציה בשם setFile המקבלת כפרמטר הפניה לקובץ בינארי ומאתחלת אותו לשמירת נתוניהם של 100 משפחות. לשם כך, יש להציב בקובץ 100 פעם, רשומה ריקה של משפחה, ובה הנתונים הבאים:
   א. מספר משפחה – 0
   ב. שם משפחה – מחרוזת ריקה באורך 20

ג. מס' נפשות במשפחה- 0
ד. מס' טלפון-0
ה. 6 סימוני "שקר" עבור רישום לחוגים.

3. כתוב/י את תת-התכניות הבאות, לצורך מערכת הרישום:

א. פונקציה להוספת משפחה בשם add. הפונקציה מקבלת כפרמטר הפניה לקובץ בינארי. הפונקציה קולטת את הנתונים עבור משפחה מתוך הקלט הסטנדרטי (מקלדת)(ללא רישום לחוגים. כלומר, יש לאתחל ב"שקר" את כל מערך הרישום לחוגים עבור משפחה חדשה). במידה שמספר המשפחה שהתקבל עדיין לא מופיע בקובץ הנתונים - יתווספו פרטי המשפחה לקובץ. במידה והמשפחה כבר מופיעה במערכת הפונקציה תזרוק את ההודעה:
"ERROR: Family is already in the file"
ועבור מספר משפחה שאינו בטווח התקין (1-100):
"ERROR: Invalid family number"

ב. פונקציה למחיקת משפחה בשם del. הפונקציה מקבלת כפרמטר הפניה לקובץ בינארי ופרמטר שני המהווה מספר משפחה. הפונקציה מוחקת מהקובץ את המשפחה שמספרה התקבל כפרמטר. (שים/י לב –משמעות המחיקה היא הצבת משפחה "ריקה" במקום, כך שהמפתח הראשי שלה הוא 0. אין צורך לדאוג לשדות האחרים). במידה והתקבל מספר משפחה שאינו בטווח התקין (1-100) הפונקציה תזרוק את הודעת החריגה:
"ERROR: Invalid family number".

ג. פונקציה לעדכון רישום בשם update. הפונקציה מקבלת כפרמטר הפניה לקובץ בינארי, פרמטר שני המהווה מספר משפחה ופרמטר שלישי תור(STL) המכיל את משפחות הממתינים לפינוי מקום בחוג. במידה ומספר המשפחה מופיע בקובץ הנתונים, הפונקציה תקלוט ערכים בוליאנים עבור 6 החוגים מהקלט הסטנדרטי. על הפונקציה לבדוק שאכן יש מקום עבור חוג נבחר (1-6) ע"י מעבר על כל המשפחות ובדיקה האם רשומים פחות מ-10 ילדים בחוג(עפ"י מס' החוג). במידה וקיים מקום הפונקציה תעדכן את רשימת החוגים הנבחרים של המשפחה שמספרה התקבל כפרמטר. במידה ולא קיים מקום, הפונקציה תוסיף את המשפחה עם ערכי true בחוג הרצוי שעבורו אין מקום פנוי לרשימת המתנה (הערה, ניתן לשמור בתור משפחה עם מס' ערכים בוליאניים, עבור בקשות לרישום למס' חוגים מלאים. כמו כן, ניתן לשמור את אותה משפחה מס' פעמים עבור כל בקשה לחוג מלא)
במידה והמשפחה אינה מופיעה במערכת, הפונקציה תזרוק את ההודעה:
"ERROR: Family is not in the file" ועבור מספר משפחה שאינו בטווח התקין (1-100): "ERROR: Invalid family number"

ד. פונקציה לטיפול ברשימת המתנה waiting. הפונקציה מקבלת כפרמטר הפניה לקובץ בינארי, פרמטר שני הפניה לתור המשפחות הממתינות לחוג, הפונקציה בודקת עבור כל המשפחות הרשומות בתור ההמתנה האם התפנה מקום בחוג אליו הן מעוניינות ברישום. לשם כך, הפונקציה תעבור על כל המשפחות הרשומות בקובץ ותבדוק עבור החוג האם מס' הרשומים סה"כ בחוג קטן מ-10(ייתכן ע"י עדכונים במהלך הרצת התכנית). במידה וכן, הפונקציה תדפיס (למסך) את שם המשפחה ואת מס' הטלפון של המשפחה, ומיד אח"כ תקלוט Y האם יש לעדכן רישום לחוג של משפחה זו או N במידה

והמשפחה לא מעוניינת כבר ברישום. במידה והקלט היה Y יש לעדכן את הרישום של המשפחה לחוג בקובץ הנתונים. בכל מקרה יש לעדכן את התור בהתאמה.

ה. פונקציה לבירור רישום לחוג בשם rishum. הפונקציה מקבלת כפרמטר הפניה לקובץ בינארי, פרמטר שני המהווה מספר משפחה, ופרמטר שלישי המהווה מספר חוג (1 עד 6). במידה ומספר המשפחה מופיע בקובץ הנתונים, ובסימון הרישום לחוג עבור משפחה זו מופיע סימון אמת, הפונקציה תחזיר אמת. אחרת – הפונקציה תחזיר שקר. במידה והמשפחה אינה מופיעה במערכת, הפונקציה תזרוק את ההודעה:
"ERROR: Family is not in the file", עבור מספר משפחה שאינו בטווח התקין (1-100): "ERROR: Invalid family number". ועבור מספר חוג שאינו בטווח התקין (1-6): "ERROR: Invalid class number".

ו. פונקציה להדפסת פרטי משפחה בשם print. הפונקציה מקבלת כפרמטר הפניה לקובץ בינארי, ופרמטר שני המהווה מספר משפחה. במידה ומספר המשפחה קיים בקובץ הנתונים, הפונקציה תדפיס לפלט הסטנדרטי (מסך) את פרטי המשפחה: שם משפחה, מס' נפשות, מס' טלפון ועבור כל חוג, אם ילד רשום לחוג יודפס Y, אחרת יודפס N. במידה והמשפחה אינה מופיעה במערכת, הפונקציה תזרוק את ההודעה:
"ERROR: Family is not in the file" ועבור מספר משפחה שאינו בטווח התקין (1-100): "ERROR: Invalid family number".

ז. פונקציה להדפסת רשימת המשפחות הרשומות בחוג מסויים בשם inClass. הפונקציה מקבלת כפרמטר הפניה לקובץ בינארי, ופרמטר שני המהווה מספר חוג (1 עד 6). הפונקציה תדפיס לפלט הסטנדרטי (מסך) את שמות המשפחות (שם משפחה) הרשומות לחוג שמספרו התקבל כפרמטר. עבור מספר חוג שאינו בטווח התקין (1-6) הפונקציה תזרוק:
"ERROR: Invalid class number".

לפניך בתוכנית ראשית הבוחנת את נכונות הפונקציות שכתבת:

```
int main()
{
	Queue<Family> q;
	fstream file;
	file.open("families.txt", ios::binary | ios::in | ios::out);
	if (!file)
	{
		cout << "ERROR: couldn't open file\n";
		return 0;
	}
	setFile (file);
	int choice;
	int snum;
	int cnum;
	cout << "Choices are:\n0 to exit\n1 to add a family\n2 to delete a
family\n3 to update rishum to classes\n4 to update waiting to classes \n5 to
check rishum for a classas\n6 to print a family\n7 to print all the families
that participate in a specific class\n";
```

```
	cout << "enter 0-7:\n";
	cin >> choice;
	while (choice)
	{
		switch (choice)
		{
		case ADD://add to the file
			try {add(file);}
			catch (const char * msg) { cout << msg; }
			break;
		case DEL://delete from file
			cout << "enter number of family to delete:\n";
			cin >> snum;
			try { del(file, snum); }
			catch (const char * msg) { cout << msg; }
			break;
		case UPDATE://update the list of classes of a family
			cout << "enter number of family to update:\n";
			cin >> snum;
			try { update(file, snum, q); }
			catch (const char * msg) { cout << msg; }
			break;
		case WAITING://update the list of classes of a waiting family
			waiting(file, q);
			break;
		case RISHUM://check rishum to a specific class
			cout << "enter number of family to check rishum:\n";
			cin >> snum;
			cout << "enter number of class to check rishum:\n";
			cin >> cnum;
			try
			{
				cout << "The family is" << (rishum(file, snum,
cnum) ? " " : " not ") << "taking the class\n";
			}
			catch (const char * msg) { cout << msg; }
			break;
		case PRINT://print the details of a specific family
			cout << "enter number of family to print:\n";
			cin >> snum;
			try { print(file, snum); }
			catch (const char * msg) { cout << msg; }
			break;
		case CLASS://print the details of all the families that are
taking a specific class
			cout << "enter number of class to check rishum:\n";
			cin >> cnum;
			try { inClass(file, cnum); }
			catch (const char * msg) { cout << msg; }
			break;
		default:
			cout << "ERROR: invalid choice\n";

		}
		cout << "\nenter 0-7:\n";
		cin >> choice;
	}
	file.close();
	return 0;
}
```