אוניברסיטת בן גוריון הפקולטה למדעי ההנדסה

המחלקה להנדסת מערכות מידע

372-1-2102 תכנות מתקדם

תשע"ח - סמסטר סתיו

תרגיל בית מס' 4 - OO CPP

תאריך הגשה: 18.01.14

הקדמה

בעבודה זו תממשו תכנה אשר תקרא ותריץ קבצי סקריפט בשפה שתוגדר במסמך זה, השפה דומה במבנה ל Python אך היא קלה יותר.
התכנית שלכם תקבל כארגומנט קובץ ותריץ את תוכנו.
דוגמא לקובץ סקריפט:

```
example_int=5# This is an int
example_bool=false# This is a bool
example_string=”abcd”# This is a string
example_int_array=[1,2,3,4]# This is an int array
example_bool_array=[false,true,false]# This is a bool array
example_string_array=[“a”,”bc”,”cde”]# This is a string array
example_int_array[0]+=10# This will set the integer on index 0 to 11
if(example_bool==true){example_int_array[2]=5}else{example_bool_array[0]=true}
if(false==true){print(“should not happen”)}
```

הגדרות כלליות לשפה

- בשפה אין רווחים, לא קיימים תווי רווח בקלט מלבד בהערות ובתוך מחרוזות.
- כל שורה מכילה פקודה אחת בלבד. בתוך תנאים יש פקודה אחת בלבד.
- תו ההפרדה בין שורות הינו '\n'

- אין בשפה שימוש בסוגריים לפעולות חשבוניות. באופן כללי אין בשפה סוגריים מקוננים, כולל סוגריים מסולסלים ומרובעים.
- כל הקלט שתקבלו הינו תקין תחבירית. אין צורך לבדוק דברים כמו רווחים, קיום סוגריים וכו'
- **<u>אין לשמור על סדר פעולות חשבון ויש לבצע את הפעולות החשבוניות משמאל לימין</u>**
- תו # מתחיל הערה, אם אתם נתקלים בתו # במהלך שורה אין משמעות למה שבא אחריו.
- כל אופרטור שמבוצע בין שני סוגים שונים של משתנים יזרוק exception מסוג BadVariableTypeException

משתנים

- בשפה ניתן להגדיר משתנים, הגדרת משתנה מתבצעת ללא הכרזה על סוגו ואת הסוג אתם מסיקים לפי הערך שמוכנס למשתנה.
- שמות משתנים יכולים להיות רק אותיות אנגליות (גדולות וקטנות) וקווים תחתונים, לדוגמא: var_name
- הגדרת משתנה תמיד תהיה באופן של name=val ללא רווחים. למשל בדוג' למעלה example_int הוא שם המשתנה וערכו 5.
- לאחר הגדרת משתנה לא ניתן לשנות את סוגו, אך כן ניתן לשנות את ערכו.
- נסיון להכניס תוכן שלא מתאים לסוגו של משתנה יוביל לזריקת exception מסוג BadVariableTypeException
- נסיון להשתמש במשתנה שלא אותחל יוביל לזריקת exception מסוג VariableNotFoundException

משתנה מסוג int

```
i=10
i=5+5
```

משתנה מסוג bool

```
i=false
```

משתנה מסוג string

```
i="abcd"
```

משתנים מסוג array

- כמו משתנים פשוטים, סוג המערך נקבע בזמן היצירה, אתחול המערך גם נקבע ביצירה וגודלו קבוע לאחר האתחול. שימו לב שלמרות שסוג וגודל המערך קבועים לאחר היצירה, תוכן המערך יכול להשתנות בזמן ריצת הסקריפט.
- נסיון להכניס תוכן שלא מתאים לסוגו של מערך יוביל לזריקת exception מסוג BadVariableTypeException
- נסיון להכניס איבר למקום לא קיים במערך יוביל לזריקת exception מסוג ArrayOutOfBoundException
- הדפסת מערך תדפיס את איבריו כשהם מופרדים בתו פסיק ולאחריו תו רווח.

```
i=[1,3,5,7]# This is an int array
k=[false,true,false]# This is a bool array
l=["a","bc","cde"]# This is a string array
```

```
i[2]=15# This will change the value on index 2 (5) to 15
k[0]=true# This will change the value on index 0 (false) to true
```

הדפסות

פקודת ההדפסה הינה print והיא מדפיסה כל סוג משתנה שהיא מקבלת, כולל מערכים. כל הדפסה תסתיים בשורה חדשה.
לדוגמא:

```
print(5)# will print “5”
print(false)# will print “false”
print(“test”)# will print “test”
print([1,2,3])# will print “1, 2, 3”
print(i)# will print the contents of i
print(1+2+3)# will print “6”
```

תנאים

בסקריפט יש אפשרות להשתמש בתנאי if, תנאי if יבדוק האם מתקיים תנאי כלשהו ויבצע <u>פקודה אחת בלבד</u> שנמצאת בתוך סוגריים מסולסלים אם כן. לדוגמא:

```
if(i==5){print(“equals”)}
if(1+5==i){print(“equals plus one”)}
```

בנוסף, ניתן להוסיף else בסוף תנאי באופן הבא:

```
if(i==5){print(“equals”)}else{print(“not equals”)}
```

סוגי אופרטורים נתמכים

עליכם לתמוך באופרטורים הבאים:

Operator symbol	Works on types	Return type
=	string, bool, int	the type of value assigned
==	string, bool, int	bool
+	int	int
+=	int	int

כל אופרטור שמבוצע בין שני סוגים שונים של משתנים יזרוק exception מסוג BadVariableTypeException

שימו לב: לטובת נוחות, **<u>אין לשמור על סדר פעולות חשבון ויש לבצע את הפעולות משמאל לימין</u>**. לדוגמא:

```
1+2==3# this equals true
3==2+1# this throws an exception because 3==2 will return bool and the subtraction cannot happen
```

חריגות - Exceptions

- עליכם להגדיר, לזרוק ולטפל בחריגות הבאות. כשחריגה מטופלת יש להדפיס את שם החריגה ואת מספר השורה בה היא התרחשה ולצאת מהתכנית.
- שלושת החריגות יבדקו. למעט חריגות אלו אינכם צריכים לטפל בשום בדיקת תקינות קלט או כל חריגה שתעלו על דעתכם, אנו לא נבדוק זאת בטסטים שנבצע.

Exception name	When is it thrown
BadVariableTypeException	When trying to compare or assign one variable type to another
VariableNotFoundException	When trying to use a variable that was not init
ArrayOutOfBoundException	When trying to access an array in a nonexisting index

הנחיות כלליות לביצוע העבודה

- התכנית שלכם מקבלת ארגומנט אחד והוא שם הקובץ אותו צריך לקרוא ולהריץ, לדוגמא: ./scriptRunner scriptfile.txt.
- צרו מחלקה כללית למשתנים ומחלקות יורשות עבור כל סוג משתנה.
- צרו אופרטורים עבור כל המשתנים שלכם.
- השקיעו מחשבה רבה לאופן שבו אתם מנתחים את הטקסט, השפה בנויה בצורה שמאפשרת קריאה נוחה משמאל לימין. מרבית פעולות הניתוח יכולות להעשות באופן רקורסיבי.
- יש צורך בתפיסת exceptions והדפסתן למסך יחד עם מספר השורה שגרמה לשגיאה ויציאה מהתכנית.
- המלצות לסדר עבודה:
  a. בנו משתני int ו bool, אופרטור השמה והדפסה. **בדקו היטב את תקינות הקוד לפני התקדמות הלאה**
  b. הוסיפו אופרטורים נוספים.
  c. בשלב זה דוגמת fib5 צריכה לעבוד.
  d. הוסיפו משתנים מסוג string
  e. הוסיפו תנאי if - בצעו בדיקות משלכם לתקינותו.
  f. הוסיפו מערכים.
  g. בשלב זה הדוגמא arrTest צריכה לעבוד.

דוגמאות לסקריפטים ופלט צפוי

סקריפט חישוב והדפסת סדרת פיבונצ'י עד האיבר 5 - שם הקובץ: fib5

```
last=1
beforelast=0
print(last)
temp=last
last+=beforelast
beforelast=temp
print(last)
temp=last
last+=beforelast
beforelast=temp
print(last)
temp=last
last+=beforelast
beforelast=temp
print(last)
temp=last
last+=beforelast
print(last)
```

פלט - תוצאת הרצת ./scriptRunner fib5

```
1
1
2
3
5
```

שימוש בתנאי ובמערכים - שם הקובץ: arrTest

```
arr=[6,6]
arr[0]=5
arr[1]+=1
if(arr[0]+2==arr[1]==false==true){print(“one”)}else{print(arr)}
```

פלט - תוצאות הרצת ./scriptRunner arrTest

```
5, 7
```

הנחיות הגשה

ההגשה בזוגות בלבד.

makefile – קובץ Make ידני שיאחד את כל הקבצים שלכם. חלק מבדיקות העבודה ייצרו את קובץ הריצה מהקוד. אין ליצור את קובץ ה Make בעזרת editor אוטומטי כלשהו, אלא לבנות אותו בעצמכם.

יש לקבץ את הקבצים לתוך תיקייה ששמה scriptRunner ולשמור את תוכנה בקובץ zip אחד ששמו מורכב ממספרי הזהות של המגישים המופרדים עם "_" בלבד.
לדוגמא,

012345678_987654321.zip

יש להגיש את הגרסה הסופית של העבודה דרך אחד המגישים בלבד , העבודה העדכנית ביותר אצל המגיש השני צריכה להיות בעלת שם שונה ממספרי הזהות של הזוג כדי למנוע בלבול בבדיקה.

התרגיל יעבור גם בדיקה אוטומטית וגם בדיקה ידנית . כדי שהתוכנית תעבור בשלום את הבדיקה האוטומטית , אין בשום אופן לשנות שמות או תוכן של קבצים שניתנו לכם.

אין לצרף קבצים נוספים, אין לתת שמות יצירתיים בעברית לקבצים שלכם, אין לשנות את השמות של הפונקציות שהוגדרו לכם.

אין צורך לבדוק תקינות של הקלט איפה שלא מוגדר.

כל ההנחיות הקודמות לסגנון כתיבת הקוד העבודה תקפות (לדוגמא: הערות באנגלית, שמות משתנים)

בעת בדיקת העבודה, אנו מפעילים מנגנון מתוחכם לזיהוי העתקות.

את העבודות יש להגיש דרך מודל.

שאלות לגבי העבודה ניתן יהיה לשאול בפורום מיוחד שייפתח באתר הקורס.