ОДСЕК ЗА СОФТВЕРСКО ИНЖЕЊЕРСТВО

АЛГОРИТМИ И СТРУКТУРЕ ПОДАТАКА 1

2019-2020

- први домаћи задатак -

Опште напомене:

1. Пре одбране сви студенти раде тест знања који се ради на рачунару коришћењем система Moodle (http://elearning.rcub.bg.ac.rs/moodle/). **Сви студенти треба да креирају налог и пријаве се на курс пре почетка лабораторијских вежби.** Пријава на курс ће бити **прихваћена и важећа** само уколико се студент региструје путем свог налога електронске поште на серверу **mail.student.etf.bg.ac.rs**.
2. Домаћи задатак 1 састоји се од два програмска проблема. Студенти проблем решавају **самостално**, на програмском језику C, Pascal или Python.
3. Реализовани програми треба да комуницирају са корисником путем једноставног менија који приказује реализоване операције и омогућава сукцесивну примену операција у произвољном редоследу.
4. Унос података треба омогућити путем читања са стандардног улаза.
5. Решења треба да буду отпорна на грешке и треба да кориснику пруже јасно обавештење у случају детекције грешке.
6. Приликом оцењивања, биће узето у обзир рационално коришћење ресурса. **Примена рекурзије се неће признати као успешно решење проблема.**
7. За све недовољно јасне захтеве у задатку, студенти треба да усвоје разумну претпоставку у вези реализације програма. Приликом одбране, демонстраторе треба обавестити која претпоставка је усвојена (или које претпоставке су усвојене) и која су ограничења програма (на пример, максимална димензија матрице и слично). Неоправдано увођење ограничавајуће претпоставке повлачи негативне поене.
8. Одбрана домаћег задатка ће се обавити у **понедељак, 02.03.2020. и уторак, 03.03.2020.** према распореду који ће накнадно бити објављен на сајту предмета.
9. За решавање задатака који имају више комбинација користити следеће формуле.

   (**R** – редни број индекса, **G** – последње две цифре године уписа):

   i = (R + G) mod 4

   j = (R + G) mod 2
10. Предаја домаћих ће бити омогућена преко Moodle система до **понедељка, 02.03.2020. у 10:00**. Детаљније информације ће бити благовремено објављене.
11. Имена датотека које се предају мора бити **dzp1.(c|pas|py)** и **dzp2.(c|pas|py)**
12. Предметни наставници задржавају право да изврше проверу сличности предатих домаћих задатака и коригују освојени број поена након одбране домаћих задатака.

24.02.2020. године *Са предмета*

Задатак 1 – линеаризација ретких вишедимензионалних низова [50 поена]

[**45 поена**] Написати интерактиван програм који илуструје рад са троугаоним матрицама. Троугаоне матрице су врста ретких матрица код којих елементи изнад или испод одређене дијагонале имају подразумевану вредност. Матрице могу бити горње или доње троугаоне. Уколико и елементи на одговарајућој дијагонали имају подразумевану вредност, онда су матрице строго горње или строго доње троугаоне. Код оваквих матрица се могу направити велике уштеде у простору уколико се памте само вредности које нису подразумеване, а матрица смешта линеаризована у виду вектора.

Зависно од редног броја проблема, саставити **један** од следећих програма, који:

0. Приказује рад са матрицом симетричном у односу на главну дијагоналу. Матрицу смештати у меморију као доње троугаону матрицу линеаризовану по врстама.
1. Приказује рад са матрицом симетричном у односу на споредну дијагоналу. Матрицу смештати у меморију као горње троугаону матрицу линеаризовану по колонама.
2. Приказује рад са строго горње троугаоном матрицом линеаризованом по колонама.
3. Приказује рад са строго доње троугаоном матрицом линеаризованом по врстама.

Програм треба да омогући ефикасан приступ произвољном елементу матрице. Незанемарљив број елемената матрице има подразумевану вредност. Те елементе матрице не треба посебно памтити, у циљу уштеде меморије. Елементима се приступа употребом адресне функције која на основу индекса елемента у матрици одређује његов индекс у вектору.

[**5 поена**] Корисник са програмом интерагује путем једноставног менија. Програм треба да испише садржај менија, а затим да чека да корисник изабере (унесе путем тастатуре) редни број неке од понуђених ставки, након чега, пре извршења, од корисника очекује да по потреби унесе додатне параметре. Поступак се понавља све док корисник у менију не изабере опцију за прекид програма. За практичну примену, корисник програма треба да има следеће могућности реализоване путем одговарајућих ставки менија:

1. стварање матрице задатих димензија уз иницијализацију неподразумеваним вредностима
2. постављање подразумеване вредности
3. дохватање задатог елемента, уз проверу валидности опсега
4. постављање вредности задатом елементу, уз проверу валидности опсега
5. дохватање броја неподразумеваних елемената
6. испис матрице (укључујући и елементе подразумеване вредности)
7. рачунање остварене уштеде меморијског простора
8. брисање матрице

За успешну реализацију програма, потребно је извести адресну функцију за приступ одговарајућем елементу матрице и ту адресну функцију искористити за приступ задатом елементу.

Напомена: у матрици је потребно чувати само неподразумеване елементе, а структуру података треба ажурирати након сваке операције промене вредности неког елемента матрице.

Задатак 2 – израчунавање вредности аритметичких израза применом стека [50 поена]

[**45 поена**] Написати програм који кориснику омогућава да израчуна вредност аритметичког израза задатог у симболичком облику у постфиксној нотацији. Корисник има могућност да унесе израз прозвољне дужине и стварне вредности симболичких променљивих (операнада) које ће бити коришћене у израчунавању. Претпоставити да се у изразу користе искључиво бинарни оператори + (сабирање), - (одузимање), * (множење) и / (дељење). Такође, претпоставити да су идентификатори свих променљивих једнословне ознаке енглеског алфабета написане велики словима.

Уколико постфиксни израз у симболичком облику није валидан за израчунавање, треба пријавити грешку одмах при уносу. Након уношења израза у симболичком облику, у сваком тренутку је могуће променити стварну вредност било које променљиве.

Зависно од редног броја проблема, направити једну од следећих имплементација стека:

0. Уланчана имплементација (једноструко уланчана листа).
1. Уланчана имплементација (једноструко уланчана кружна листа).

[**5 поена**] Корисник са програмом интерагује путем једноставног менија. Програм треба да испише садржај менија, а затим да чека да корисник изабере (унесе путем тастатуре) редни број неке од понуђених ставки, након чега, пре извршења, од корисника очекује да по потреби унесе додатне параметре. Поступак се понавља све док корисник у менију не изабере опцију за прекид програма.

Додатак – имплементација уланчане листе на различитим програмским језицима

У прилогу се налазе примери дефиниције чвора и заглавља једноструко уланчане листе целих бројева на различитим програмским језицима. Дате дефиниције се могу користити као узор за реализацију студентских решења.

Програмски језик C

```
typedef struct listNode {
     int info;
     struct listNode *next;
} ListNode;
typedef struct listHeader {
     ListNode *head, *tail;
     int numElem;
} ListHeader;
```

Програмски језик Pascal

```
TYPE pNode = ^ListNode;
ListNode = RECORD
     info: integer;
     next: pNode;
     END;
ListHeader = RECORD
     head, tail: pNode;
     numElem: integer;
     END;
```

Програмски језик Python

```
class ListNode:
    def __init__(self, info=None):
        self.info = info
        self.next = None
class ListHeader:
    def __init__(self):
        self.head = None
        self.tail = None
        self.numElem = None
```