Descrizione del corso

LEZIONI UniStudium ONLINE

Controlla orari e appuntamenti online sulla tua email o sugli Annunci del Corso

1) clicca qui per collegamento a TEAMS

2) clicca qui per Lezione Online

Ricerca nei forum

Vai

Ricerca avanzata

Annunci recenti

10 mar 2020, 15:26:06
Stefano MARCUGINI
Inizio lezioni on-line mercoledì 11 marzo ore 11
Argomenti precedenti ...

Prossimi eventi

Non ci sono eventi prossimi
Vai al calendario...

Attività recente

Attività a partire da martedì, 10 marzo 2020, 11:46
Report completo dell'attività recente...

Aggiornamenti del corso:

Aggiunto URL
Prima lezione: campi e metodi statici, uguaglianza oggetti

Aggiunto File
Esercitazione 1: Vettore 2020

Nuovi interventi nel forum:

Stefano MARCUGINI 10 mar 2020, 15:26:06
Inizio lezioni on-line mercoledì 11 marzo ore 11

Navigazione

- Dashboard
- Home del sito
- Pagine del sito
- I miei corsi
  - ANALISI MATEMATICA - MODULO I (2019/20)
  - ANALISI MATEMATICA - MODULO II (2019/20)
  - ARCHITETTURA DEGLI ELABORATORI CON LABORATORIO - A...
  - ARCHITETTURA DEGLI ELABORATORI CON LABORATORIO - M...
  - MATEMATICA DISCRETA - MODULO I (2019/20)
  - MATEMATICA DISCRETA - MODULO II (2019/20)
  - PROGRAMMAZIONE I CON LABORATORIO (2019/20)
  - PROGRAMMAZIONE II CON LABORATORIO (2019/20)
    - Partecipanti
    - Badge
    - Competenze
    - Valutazioni
    - Introduzione
    - Argomento 1
    - Argomento 2
    - Argomento 3
    - Argomento 4
    - Argomento 5
    - Argomento 6
    - Argomento 7
    - Argomento 8
    - Argomento 9
    - Argomento 10
    - Argomento 11
    - Argomento 12
    - Argomento 13
    - Argomento 14
    - Argomento 15
    - Argomento 17
    - Argomento 18
    - Argomento 19
    - Argomento 20
    - Argomento 21
    - Argomento 22
    - Argomento 23
  - ANALISI MATEMATICA - MODULO I (2018/19)
  - ARCHITETTURA DEGLI ELABORATORI CON LABORATORIO - A...
  - Altro...

Amministrazione

- Amministrazione del corso
  - Disiscrivimi da 19_173013

Attività

- Forum
- Lezioni
- Risorse

I miei corsi

- 19_173011 ANALISI MATEMATICA - MODULO I (2019/20)
- 19_173012 ANALISI MATEMATICA - MODULO II (2019/20)
- 19_173015 ARCHITETTURA DEGLI ELABORATORI CON LABORATORIO - ARCHITETTURA DEGLI ELABORATORI - MOD I (2019/20)
- 19_173016 ARCHITETTURA DEGLI ELABORATORI CON LABORATORIO - MOD. 2 (2019/20)
- 19_173020 MATEMATICA DISCRETA - MODULO I (2019/20)
- 19_173021 MATEMATICA DISCRETA - MODULO II (2019/20)
- 19_173018 PROGRAMMAZIONE I CON LABORATORIO (2019/20)
- 19_173013 PROGRAMMAZIONE II CON LABORATORIO (2019/20)
- 18_157519 ANALISI MATEMATICA - MODULO I (2018/19)
- 18_157528 ARCHITETTURA DEGLI ELABORATORI CON LABORATORIO - ARCHITETTURA DEGLI ELABORATORI - MOD I (2018/19)
- 18_157523 MATEMATICA DISCRETA - MODULO I (2018/19)
- 18_157526 PROGRAMMAZIONE I CON LABORATORIO (2018/19)
- 18_157521 PROGRAMMAZIONE II CON LABORATORIO (2018/19)
- 17_125891 DIRITTO PRIVATO [Cognomi M-Z] (2017/18)
- 17_125220 ECONOMIA AGRARIA (2017/18)
- 17_74080 ECONOMIA E GESTIONE DELLE IMPRESE (2017/18)
- 17_125030 INFORMATICA (2017/18)
- 17_128522 MATEMATICA FINANZIARIA [Cognomi M-Z] (2017/18)
- 17_127970 MICROECONOMIA [Cognomi M-Z] (2017/18)
- 17_83076 PROGRAMMAZIONE DICHIARATIVA (2017/18)
- 17_125599 PROGRAMMAZIONE II (2017/18) [M]
- 17_127986 RAGIONERIA GENERALE ED APPLICATA [Cognomi M-Z] (2017/18)
- 17_128066 STATISTICA [Cognomi M-Z] (2017/18)
- 16_6957 Diritto commerciale (2016/17)
- 16_7123 Diritto privato (2016/17)
- 16_6965 Economia agraria (2016/17)
- 16_7128 Economia aziendale (2016/17)
- 16_6959 Economia e gestione delle imprese (2016/17)
- 16_7095 Matematica generale (2016/17)
- 16_7133 Microeconomia (2016/17)
- 16_6962 Ragioneria generale ed applicata (2016/17)
- 15_157 Microeconomia (2015/16)

LEZIONI UniStudium ONLINE ☐

Controlla orari e appuntamenti online sulla tua email o sugli Annunci del Corso

1) clicca qui per collegamento a TEAMS

2) clicca qui per Lezione Online

Annunci

La correzione del compito prevista per mercoledì 11 marzo è rinviata a data da destinarsi. ☐

- Programma del corso
- Testi di riferimento
- Siti di riferimento
- Modalità di esame

Lucidi delle lezioni:

- Parte 1
- Parte 1 bis: Complessita
- Parte 1 ter: Ricorsione
- Parte 2
- Parte 3
- Parte 4
- Parte 5
- Documentazione JAVADOC
  - Lucidi Javadoc
  - Esempio Javadoc

Argomento 1

Esercitazioni di approfondimento

Lezione Incapsulamento ☐
Not attempted

Lezione Passaggio di Parametri ☐
Not attempted

Lezione Sincronizzazione ☐
Not attempted

Esercizi

Esercizi

Video delle lezioni on-line

Prima lezione: campi e metodi statici, uguaglianza oggetti ☐

Purtroppo il video si blocca dopo mezz'ora, l'audio è completo.

Esercitazioni 2020

Esercitazione 1: Vettore 2020

Definire una classe che rappresenta un vettore di interi

1) definire un costruttore che dati n e m istanzia un vettore di n elementi minori di m generati casualmente

2) definire un metodo che visualizza il contenuto del vettore

3) definire un metodo che calcola la media degli elementi

4) definire un metodo che dato n restituisce la posizione di n nel vettore oppure -1 se n non è presente nel vettore (3 versioni: con ciclo for e con ciclo while e ricorsiva)

5) definire un metodo che dato n restituisce il numero di volte che n è presente nel vettore (2 versioni: iterativa e ricorsiva)

6) definire un metodo che restituisce il massimo del vettore (2 versioni: iterativa e ricorsiva)

Esercitazioni 2019

Esercitazione 1: numero razionale 2019

Si definisca una classe per rappresentare un numero razionale

Si definiscano funzioni per:
calcolare il Massimo Comun Divisore fra due interi
(usando l'algoritmo euclideo:
se due numeri son uguali il MCD è uno dei due; altrimenti si sostituisca il più grande con la loro differenza finché diventano uguali)
leggere un intero positivo da tastiera

Poi:

definiamo un altro costruttore a 2 parametri

definiamo un costruttore a 1 parametro

Definiamo un metodo che restituisce un razionale chiedendo all'utente numeratore e denominatore (non zero) da tastiera

definiamo metodi per:

ridurre un razionale ai minimi termini

modificare l'oggetto moltiplicandolo per un altro numero razionale

restituire un nuovo oggetto che rappresenta il prodotto con un altro numero razionale

calcolare il minimo comune multiplo di 2 interi (prodotto diviso MCD)

modificare l'oggetto sommandolo ad un altro numero razionale

restituire un nuovo oggetto che rappresenta la somma con un altro numero razionale

Esercitazione 2: Vettore 2019

Definire una classe che rappresenta un vettore di interi

1) definire un costruttore che dati n e m istanzia un vettore di n elementi minori di m generati casualmente

2) definire un metodo che visualizza il contenuto del vettore

3) definire un metodo che calcola la media degli elementi

4) definire un metodo che dato n restituisce la posizione di n nel vettore oppure -1 se n non è presente nel vettore (3 versioni: con ciclo for e con ciclo while e ricorsiva)

5) definire un metodo che dato n restituisce il numero di volte che n è presente nel vettore (2 versioni: iterativa e ricorsiva)

6) definire un metodo che dato il massimo del vettore (2 versioni: iterativa e ricorsiva)

Esercitazione Lista 2019

1) definire una lista concatenata di interi
2) definire un metodo costruttore che, dato n, istanzia una lista di n elementi casuali
3) definire un metodo di visualizzazione (versione iterativa con while, versione iterativa con for, versione ricorsiva)
4) definire un metodo che restituisce la somma degli elementi (versione iterativa, versione ricorsiva)
5) definire un metodo che, dato un parametro n, restituisce il numero di elementi presenti nella lista più grandi di n (compito 23/06/2015)

6) definire il metodo verifica che, in modo efficiente, verifica se ogni elemento della lista è maggiore dei precedenti (la proprietà si considera sempre vera per il primo elemento della lista) (compito 16/01/2016).

7) definire il metodo verifica che, in modo efficiente, verifica se un elemento è presente nella lista (versione iterativa e ricorsiva).

8) Definire Il metodo copia che, dato come parametro un intero N, restituisce una nuova lista formata
dagli elementi della lista maggiori di N.
La lista originale non deve essere modificata. L'ordine degli elementi presenti nella nuova
lista può essere diverso dall'ordine in cui si trovavano nella lista originale.
(versione iterativa che inverte l'ordne degli elementi)
(versione ricorsiva che mantiene l'ordine degli elementi)

(compito 23 giugno 2009)

9) Derfinire il metodo maxDivisibili che dato n restituisce il massimo degli elementi della lista divisibili per n (si suppongano tutti gli elementi non negativi). Versione iterativa e ricorsiva.

10) Derfinire il metodo aggiorna che dato n sostituisce ogni elemento divisibile per n con la sua posizione. Versione iterativa e ricorsiva.

Esercitazione Veicoli (Esonero 19 aprile 2016)

Si definisca una gerarchia di classi per rappresentare i veicoli di una compagnia di trasporti. I veicoli sono di due tipi: camion e autobus. Per ogni veicolo si rappresenti la potenza (in kW) e i chilometri percorsi.
I chilometri percorsi devono essere modificabili solo tramite un metodo viaggio che, dato un parametro intero n, incrementa i chilometri percorsi solo se n è positivo.
Per i camion si rappresenti il fatto che abbiano o meno un rimorchio,
mentre per gli autobus si rappresenti il numero di passeggeri.
Inoltre per camion ed autobus si definisca un metodo che restituisce il costo annuale dell'assicurazione: per i camion è la potenza per 1000, più 500 se hanno il rimorchio, più 500 se hanno percorso meno di 50000 chilometri, mentre per gli autobus è la potenza per 750, più 1000 se hanno più di 50 passeggeri, più 200 se hanno percorso più di 20000 chilometri.
Si definisca una classe Garage che contiene come campo un array V in cui è possibile inserire sia Camion che Autobus.

Si definisca inoltre un costruttore della classe Garage che ha due parametri n e m e che istanzia il vettore V con n camion ed m autobus.
Si definiscano i campi degli oggetti in modo casuale.

Si definisca infine un metodo della classe Garage che, dato un parametro *n* verifica se *n* euro sono sufficienti ad assicurare i veicoli presenti in V.

Esercitazione su liste di stringhe

Esercizio (Esonero 20 aprile 2018)
Si definisca una classe Lista che rappresenta una lista di stringhe concatenata tramite riferimenti.
Nella classe Lista si definiscano:
• un costruttore il costruttore che, dato un parametro intero n, costruisce la lista formata dalle stringhe "Elemento1", "Elemento2, ... , "Elementon", in questo ordine.
• il metodo iterativo verifica che, dati un intero n, ed una stringa s verifica se nella lista sono presenti almeno n stringhe uguali ad s.
• la versione ricorsiva del metodo verifica.
Si realizzi infine una applicazione che:
• crea una lista di 100 elementi
• verifica se nella lista sono presenti almeno 10 elementi uguali a "casa" usando il metodo iterativo.
• verifica se nella lista sono presenti almeno 5 elementi uguali a "Elemento3" usando il metodo ricorsivo.

Parte seconda (6 settembre 2017)

Nella classe Lista si aggiungano:
• il metodo inserisci che dato un parametro parola di tipo stringa inserisce la parola in testa alla lista.
• il metodo conta che, dato un parametro n, restituisce il numero di stringhe contenute nella lista la cui lunghezza è diversa da n.
• la versione ricorsiva del metodo conta.

Nel main si aggiungano chiamate di metodo che:

• inseriscono nella lista le parole "gatto" , "topo" ed "elefante" usando tre volte il metodo inserisci;
• stampano a video il risultato della chiamata del metodo conta iterativo con parametro uguale a 5.
• stampano a video il risultato della chiamata del metodo conta ricorsivo con parametro uguale a 6.

Esercitazione musicisti

Esercizio (Esonero 20 aprile 2015)
Si definisca una gerarchia di classi per rappresentare i musicisti di una orchestra.

I musicisti sono due tipi: professionisti e dilettanti.

Per ogni musicista si rappresenti: il nome e l'età.

L'età non deve essere modificabile liberamente, ma solo attraverso un metodo *compleanno()*, disponibile a chiunque usi un oggetto musicista, che ne incrementa il valore di 1.

Inoltre per i musicisti professionisti si rappresenti il numero di anni frequentati al conservatorio,

mentre per i musicisti dilettanti si rappresenti se sanno leggere o no la musica.

Per entrambe le classi si definisca un metodo che restituisce il *rimborsoSpese*.

Per i musicisti professionisti il rimborso è 10 per il numero di anni frequentati al conservatorio, più 10 se l'età è maggiore di 35;

per i musicisti dilettanti il rimborso è 2 per l'età, più 10 se sanno leggere la musica.

Si definisca una classe Archivio che contiene come campo un array V in cui è possibile inserire sia musicisti professionisti che musicisti dilettanti.

Si definisca inoltre un costruttore della classe Archivio che ha due parametri *n* e *m* e che istanzia il vettore V con *n* musicisti professionisti ed *m* musicisti dilettanti.

Si definiscano i campi degli oggetti in modo casuale.

Si definisca infine un metodo della classe Archivio che, dato un parametro *n*, restituisce il numero di musicisti il cui costo di rimborso supera *n*.

Esercitazione Corsi

Esercizio (Esonero 20 aprile 2015)
Si definisca una gerarchia di classi per rappresentare i corsi erogati da una agenzia formativa.

I corsi sono due tipi: di base e avanzati.

Per ogni corso si rappresenti: il numero di partecipanti e la durata in ore.

Il numero di partecipanti, una volta assegnato dal costruttore, non deve essere modificabile liberamente, ma solo attraverso un metodo *iscrivi*. Il metodo *iscrivi*, disponibile a chiunque usi un oggetto corso, dato un parametro n, incrementa il numero di partecipanti di *n*.

Inoltre per i corsi di base si rappresenti il numero di ore di laboratorio,

mentre per i corsi avanzati si rappresenti se è previsto o no un tirocinio.

Per entrambe le classi si definisca un metodo che restituisce il *costoIscrizione*.

Per i corsi di base il *costoIscrizione* è 1000 più 500 se ci sono meno di 100 iscritti, più 500 se sono previste più di 50 ore di laboratorio;

per i corsi avanzati il *costoIscrizione* è 2000 più 500 se ci sono meno di 50 iscritti, più 500 se è previsto un tirocinio.

Si definisca una classe Segreteria che contiene come campo un array V in cui è possibile inserire sia corsi di base che corsi avanzati.

Si definisca un costruttore della classe Segreteria che ha due parametri *n* e *m* e che istanzia il vettore V con *n* corsi di base ed *m* corsi avanzati. Si definiscano i campi degli oggetti in modo casuale.

Si definisca infine un metodo della classe Segreteria che, dato un parametro *n*, restituisce il numero di corsi il cui *costoIscrizione* è inferiore ad *n*.

Esempio quicksort

Esempio dell'algoritmo di ordinamento quicksort;

versione ricorsiva.

Esempio applicazione grafica

scriviamo una applicazione grafica costituita da un bottone e una label;

cliccando sul bottone deve avvenire:
sulla label compare il numero di volte che il bottone è stato premuto dall'inizio dell'applicazione

Esempio applicazione grafica 1

scriviamo una applicazione grafica costituita da due campi di testo destinati a contenere valori interi, un pannello e un bottone;

cliccando sul bottone deve avvenire: se i valori contenuti nei campi di testo son uguali il pannello si colora di verde, altrimenti di rosso, se uno dei due non contiene un numero il pannello si colora di giallo

Esempio applicazione grafica 2

Si definisca una applicazione grafica costituita da una finestra che contiene (in questo ordine):

una label contenente il valore 0 ed un bottone.

Cliccando sul bottone deve avvenire:

inizialmente il valore della label viene incrementato di 1;

una volta raggiunto il valore 10 il valore della label viene decrementato di 1 fino a quando si torna al valore 0;

e così via alternando 10 incrementi e 10 decrementi di 1.

Esempio applicazione grafica Gioco Fagioli

Si definisca una applicazione grafica che permetta all'utente di giocare il seguente gioco:

L'applicazione sceglie un numero, l'utente lo deve indovinare.

Ad ogni tentativo, l'applicazione informa l'utente se il valore proposto è più grande o più piccolo del valore da indovinare.

In caso di vittoria viene comunicato il numero di tentativi utilizzati.

Esercitazione su file 2019

Esercitazione su file

Si scriva una applicazione che contiene i seguenti metodi:

Metodo 0

Si scriva un metodo che dato il nome di un file e due interi n e m,

scrive in un file di caratteri con quel nome n interi compresi fra 0 e m, uno per ogni riga.

Metodo 1 (esonero 03 giugno 2015)
Si consideri un file di caratteri che contiene un numero intero per ogni riga.
Si scriva un metodo che dato il nome del file ed un intero N restituisce la somma dei valori contenuti nel file che sono maggiori di N.

Metodo 2 (esonero 1 giugno 2016)
Si consideri un file di caratteri.
Si scriva un metodo che dato il nome del file restituisce il numero delle righe del file che non rappresentano numeri interi.

Metodo 3 (esonero 7 giugno 2017)
Si consideri un file di caratteri.

Si scriva una applicazione che contiene un metodo che, dato il nome del file di caratteri, verifica se il file contiene almeno una linea che rappresenta un numero.

Metodo 4

Si scriva un metodo che dato il nome di un file e due interi n e m,

scrive in un file binario con quel nome n interi compresi fra 0 e m, uno per ogni riga.

Metodo 5 (esonero 03 giugno 2015)

Si consideri un file binario che contiene un numero intero per ogni riga.

Si scriva un metodo che dato il nome del file ed un intero N restituisce la somma dei valori contenuti nel file che sono maggiori di N.

Si scriva anche il metodo main che richiama i suddetti metodi e ne stampa a video il risultato.

Esercitazione applicazione client-server, compito 30/06/2016

Esercitazione applicazione client-server, compito 30/06/2016.

Si definisca una applicazione client server basata su socket.
Il server deve poter rispondere contemporaneamente a più client.
Il server genera casualmente un intero N (lo stesso valore è usato per tutti i client).
Il server invia poi al client N stringhe: "stringa1", "stringa2", ..., "stringaN".
La trasmissione è terminata inviando la stringa "BYE".
Alla fine della trasmissione il client stampa sullo schermo la somma delle lunghezze delle stringhe ricevute.
Se necessario, si gestiscano le problematiche di sincronizzazione.

Esercitazione client server esonero 2016

Esercitazione client server esonero 2016

Si definisca una applicazione client server basata su socket.
Il server deve poter rispondere contemporaneamente a più client.
Il client genera casualmente un intero N. Il client invia poi al server N valori numerici interi positivi generati casualmente.
La fine della sequenza viene segnalata inviando il valore numerico -1.
Il server genera casualmente un intero M (lo stesso valore è usato per tutti i client).
Ogni volta che riceve un numero divisibile per M, il server scrive a video il numero complessivo di valori divisibili per M ricevuti finora da tutti i client ed il numero di valori non divisibile per M ricevuti solo dal proprio client.
Se necessario, si gestiscano le problematiche di sincronizzazione.

Esercitazione client server compito 22 febbraio 2019

Si definisca una applicazione client server basata su socket.

Il server deve poter rispondere contemporaneamente a più client.

Il client invia al server 1000 interi compresi fra 1 e 1000. La trasmissione è terminata inviando il valore -1.

Il server non sa che riceverà dai client esattamente 1000 valori. Ogni volta che un client si disconnette il server scrive a video: il numero di valori pari ed il numero di valori dispari ricevuti finora da tutti i client; inoltre scrive anche la differenza ed il rapporto fra queste 2 quantità.

Se necessario, si gestiscano le problematiche di sincronizzazione.

Esercitazioni 2018

Esercitazione 1: numero razionale 2018

Si definisca una classe per rappresentare un numero razionale

Si definiscano funzioni per:
calcolare il Massimo Comun Divisore fra due interi
(usando l'algoritmo euclideo:
se due numeri son uguali il MCD è uno dei due; altrimenti si sostituisca il più grande con la loro differenza finché diventano uguali)
leggere un intero positivo da tastiera

Poi:

definiamo un costruttore che chiede all'utente numeratore e denominatore (non zero)
definiamo un altro costruttore a 2 parametri

definiamo metodi per:
ridurre un razionale ai minimi termini
modificare l'oggetto moltiplicandolo per un altro numero razionale
restituire un nuovo oggetto che rappresenta il prodotto con un altro numero razionale
calcolare il minimo comune multiplo di 2 interi (prodotto diviso MCD)

modificare l'oggetto sommandolo ad un altro numero razionale
restituire un nuovo oggetto che rappresenta la somma con un altro numero razionale

Esercitazione 2: Vettore 2018

Definire una classe che rappresenta un vettore di interi

1) definire un costruttore che dati n e m istanzia un vettore di n elementi minori di m generati casualmente

2) definire un metodo che visualizza il contenuto del vettore

3) definire un metodo che calcola la media degli elementi

4) definire un metodo che dato n restituisce la posizione di n nel vettore oppure -1 se n non è presente nel vettore (3 versioni: con ciclo for e con ciclo while e ricorsiva)

5) definire un metodo che dato n restituisce il numero di volte che n è presente nel vettore (2 versioni: iterativa e ricorsiva)

Esercitazione Lista 2018

1) definire una lista concatenata di interi
2) definire un metodo costruttore che, dato n, istanzia una lista di n elementi casuali
3) definire un metodo di visualizzazione (versione iterativa con while, versione iterativa con for, versione ricorsiva)
4) definire un metodo che restituisce la somma degli elementi (versione iterativa, versione ricorsiva)
5) definire un metodo che, dato un parametro n, restituisce il numero di elementi presenti nella lista più grandi di n (compito 23/06/2015)

6) definire il metodo verifica che, in modo efficiente, verifica se ogni elemento della lista è maggiore dei precedenti (la proprietà si considera sempre vera per il primo elemento della lista) (compito 16/01/2016).

7) definire il metodo verifica che, in modo efficiente, verifica se un elemento è presente nella lista (versione iterativa e ricorsiva).

8) Definire Il metodo copia che, dato come parametro un intero N, restituisce una nuova lista formata
dagli elementi della lista maggiori di N.
La lista originale non deve essere modificata. L'ordine degli elementi presenti nella nuova
lista può essere diverso dall'ordine in cui si trovavano nella lista originale.
(versione iterativa che inverte l'ordne degli elementi)
(versione ricorsiva che mantiene l'ordine degli elementi)

(compito 23 giugno 2009)

9) Derfinire il metodo maxDivisibili che dato n restituisce il massimo degli elementi della lista divisibili per n (si suppongano tutti gli elementi non negativi). Versione iterativa e ricorsiva.

10) Derfinire il metodo aggiorna che dato n sostituisce ogni elemento divisibile per n con la sua posizione. Versione iterativa e ricorsiva.

Esercitazione Veicoli (Esonero 19 aprile 2016)

Si definisca una gerarchia di classi per rappresentare i veicoli di una compagnia di trasporti. I veicoli sono di due tipi: camion e autobus. Per ogni veicolo si rappresenti la potenza (in kW) e i chilometri percorsi.
I chilometri percorsi devono essere modificabili solo tramite un metodo viaggio che, dato un parametro intero n, incrementa i chilometri percorsi solo se n è positivo.
Per i camion si rappresenti il fatto che abbiano o meno un rimorchio,
mentre per gli autobus si rappresenti il numero di passeggeri.
Inoltre per camion ed autobus si definisca un metodo che restituisce il costo annuale dell'assicurazione: per i camion è la potenza per 1000, più 500 se hanno il rimorchio, più 500 se hanno percorso meno di 50000 chilometri, mentre per gli autobus è la potenza per 750, più 1000 se hanno più di 50 passeggeri, più 200 se hanno percorso più di 20000 chilometri.
Si definisca una classe Garage che contiene come campo un array V in cui è possibile inserire sia Camion che Autobus.

Si definisca inoltre un costruttore della classe Garage che ha due parametri n e m e che istanzia il vettore V con n camion ed m autobus.
Si definiscano i campi degli oggetti in modo casuale.

Si definisca infine un metodo della classe Garage che, dato un parametro *n* verifica se *n* euro sono sufficienti ad assicurare i veicoli presenti in V.

Esercitazione Lista (Esonero 19 aprile 2016) + Pila e Coda di interi

Esercizio 2
Si definisca la classe Lista che rappresenta una lista di interi concatenata tramite riferimenti.
Nella classe Lista
si definisca:
•un metodo costruttore che, dato un parametro n costruisce una lista di n valori casuali non negativi.

• un metodo iterativo che, dato un parametro n, restituisce il massimo degli elementi divisibili per n.

•la versione ricorsiva del metodo sopra

Si realizzi infine una applicazione che:

• crea una lista di 100 elementi
• stampa a video il massimo degli elementi della lista divisibili per 5.

Pila (di interi)

collezione di dati in cui i dati vengono inseriti ed estratti

LIFO Last In First Out (gli ambienti delle chiamate vengono gestiti con questa strategia)

lista con inserimenti in testa ed estrazioni in testa.

---

coda (di interi)

collezione di dati in cui i dati vengono inseriti ed estratti

FIFO First In First Out

lista con inserimenti in coda ed estrazioni in testa.

Esercitazione Lista (Compito 15 novembre 2017)

Compito 15 novembre 2017

Si definisca la classe *Lista* che rappresenta una lista di stringhe concatenata tramite riferimenti.

Nella classe *Lista* si definisca:

il costruttore che, dato un parametro intero *n*, costruisce la lista formata dalle stringhe "Elemento1", "Elemento2, ... , "Elementon", in questo ordine.
il metodo *ennesimo* che, dato un parametro *n*, restituisce la stringa contenuta nell'elemento di posizione *n* nella lista (si inizi a contare da 1);

se la lista non contiene almeno *n* elementi o se *n* non è positivo si restituisca null.

la versione ricorsiva del metodo *ennesimo*.

Si realizzi infine una applicazione che:

crea una lista di 10 elementi;
stampa a video il risultato della chiamata del metodo *ennesimo* iterativo con parametro uguale a 5.
stampa a video il risultato della chiamata del metodo *ennesimo* ricorsivo con parametro uguale a 15.

Esercitazione Veicoli (Compito 16 gennaio 2018)

Compito 16 gennaio 2018

Si definisca una gerarchia di classi per rappresentare gli pneumatici venduti da un gommista.

Gli pneumatici sono di due tipi: estivi e invernali. Per ogni pneumatico si rappresenti le marca e la durata garantita (in chilometri).

Per gli pneumatici estivi si rappresenti anche la temperatura minima di utilizzo, mentre per gli pneumatici invernali si rappresenti il fatto che siano utilizzabili tutto l'anno oppure no.

Inoltre per gli pneumatici estivi e per quelli invernali si definisca un metodo che ne calcola il costo:

per quelli estivi il costo è 0.005 per i chilometri garantiti più 5 se utilizzabili sotto i 15 gradi,

mentre per quelli invernali il costo è è 0.007 per i chilometri garantiti più 7 se utilizzabili tutto l'anno.

Si definisca una classe Negozio che contiene come campo un array V in cui è in cui è possibile inserire sia pneumatici estivi che pneumatici invernali.

Si definisca inoltre un costruttore della classe Negozio che ha due parametri n e m e che istanzia il vettore V con n pneumatici estivi ed m pneumatici invernali. Si definiscano i campi degli oggetti in modo casuale.

Si definisca infine un metodo della classe Negozio che, dato il nome di una marca ed un valore *n,* restituisce il numero degli pneumatici di quella marca presenti nel vettore V che costano meno di *n*.

Esempio applicazione grafica

scriviamo una applicazione grafica costituita da un bottone e una label;

cliccando sul bottone deve avvenire:
sulla label compare il numero di volte che il bottone è stato premuto dall'inizio dell'applicazione

Esempio applicazione grafica 2

scriviamo una applicazione grafica costituita da due campi di testo, un pannello e un bottone;

cliccando sul bottone deve avvenire:
se i valori contenuti nei campi di testo son uguali il pannello si colora di verde, altrimenti di rosso

Esempio applicazione grafica 3

Si definisca una applicazione grafica costituita da una finestra che contiene (in questo ordine):

una label contenente il valore 0 ed un bottone.

Cliccando sul bottone deve avvenire:

inizialmente il valore della label viene incrementato di 1;

una volta raggiunto il valore 10 il valore della label viene decrementato di 1 fino a quando si torna al valore 0;

e così via alternando 10 incrementi e 10 decrementi di 1.

Esempio applicazione grafica 4

scriviamo una applicazione grafica costituita da due campi di testo destinati a contenere valori interi, un pannello e un bottone;

cliccando sul bottone deve avvenire: se i valori contenuti nei campi di testo son uguali il pannello si colora di verde, altrimenti di rosso, se uno dei due non contiene un numero il pannello si colora di giallo

Esercitazione su file 2018

Esercitazione su file

Si scriva una applicazione che contiene i seguenti metodi:

Metodo 1 (esonero 03 giugno 2015)
Si consideri un file di caratteri che contiene un numero intero per ogni riga.
Si scriva un metodo che dato il nome del file ed un intero N restituisce la somma dei valori contenuti nel file che sono maggiori di N.

Metodo 2 (esonero 1 giugno 2016)
Si consideri un file di caratteri.
Si scriva un metodo che dato il nome del file restituisce il numero delle righe del file che non rappresentano numeri interi.

Metodo 3 (esonero 7 giugno 2017)
Si consideri un file di caratteri.

Si scriva una applicazione che contiene un metodo che, dato il nome del file di caratteri, verifica se il file contiene almeno una linea che rappresenta un numero.

Si scriva anche il metodo main che richiama il suddetto metodo e ne stampa a video il risultato.

Esercitazione applicazione client-server, compito 30/06/2016

Esercitazione applicazione client-server, compito 30/06/2016.

Si definisca una applicazione client server basata su socket.
Il server deve poter rispondere contemporaneamente a più client.
Il server genera casualmente un intero N (lo stesso valore è usato per tutti i client).
Il server invia poi al client N stringhe: "stringa1", "stringa2", ..., "stringaN".
La trasmissione è terminata inviando la stringa "BYE".
Alla fine della trasmissione il client stampa sullo schermo la somma delle lunghezze delle stringhe ricevute.
Se necessario, si gestiscano le problematiche di sincronizzazione.

Esercitazione client server esonero 2016

Esercitazione client server esonero 2016

Si definisca una applicazione client server basata su socket.
Il server deve poter rispondere contemporaneamente a più client.
Il client genera casualmente un intero N. Il client invia poi al server N valori numerici interi positivi generati casualmente.
La fine della sequenza viene segnalata inviando il valore numerico -1.
Il server genera casualmente un intero M (lo stesso valore è usato per tutti i client).
Ogni volta che riceve un numero divisibile per M, il server scrive a video il numero complessivo di valori divisibili per M ricevuti finora da tutti i client.
Se necessario, si gestiscano le problematiche di sincronizzazione.

Esercitazione applicazione client-server, esonero 2017

Esercitazione applicazione client-server, esonero 2017

Si definisca una applicazione client server basata su socket.

Il server deve poter rispondere contemporaneamente a più client.

Il client genera casualmente un intero N.

Il client invia poi al server N stringhe: "stringa1", "stringa2", ..., "stringaN".

La trasmissione è terminata inviando la stringa "BYE".

Ogni volta che riceve una stringa più lunga di quelle ricevute fino a quel momento da tutti i client, il server la scrive a video.

Se necessario, si gestiscano le problematiche di sincronizzazione.

Argomento 2

Esami

- Prova di esame 29 maggio 2007
- Esame 18 giugno 2007
- Esame 9 luglio 2007
- Esame 3 settembre 2007
- Esame 18 settembre 2007
- Esame 22 gennaio 2008
- Risutati esame 11 febbraio
- Esame 11 febbraio 2008
- Esame 26 giugno 2008
- Esame 15 luglio 2008
- Esame 2 settembre 2008
- Soluzione esame 20 gennaio 2009
- Testo Compito 23 giugno 2009
- Compito 25 settembre 2009
- Compito 27 gennaio 2010
- Compito 11 febbraio 2010
- Testo compito 16 giugno 2010
- Testo compito 9 luglio 2010
- Testo compito 16 settembre 2010
- Testo compito 30 settembre 2010
- Testo compito 16 febbraio 2011
- Testo compito 2 marzo 2011
- Testo Compito 16 giugno 2011
- Testo Compito 5 luglio 2011
- Testo Compito 6 settembre 2011
- TestoCompito26settembre2011.pdf
- TestoCompito17Gennaio2012.pdf
- Testo compito 8 giugno 2012
- Testo Compito 12 luglio 2012
- Testo Compito 11 settembre 2012
- Testo Compito 27 settembre 2012
- Testo Compito 16 gennaio 2013
- Testo Compito 7 febbraio 2013
- Testo Compito 13 giugno 2013
- Testo Compito 16 luglio 2013
- Testo Compito 5 settembre 2013
- Testo Compito 20 settembre 2013
- Testo Compito 20 gennaio 2014
- Testo Compito 11 febbraio 2014
- Testo Compito Esonero 24 Aprile 2014
- Testo Compito Esonero 04 giugno 2014
- Testo Compito 23 giugno 2014
- Testo Compito 21 luglio 2014
- Testo Compito 03 settembre 2014
- Testo Compito 24 settembre 2014
- Testo Compito 06 febbraio 2015
- Testo Compito 20 febbraio 2015
- Testo Compito Esonero 20 aprile 2015
  - Soluzione primo compito di esonero 2015
- Testo compito 23 giugno 2015
- Compiti recenti 2015
- Compiti 2016
- Testo compito esonero 19 aprile 2016
  - Soluzione primo esonero 2016
- Compiti 2017
- Testo compito esonero 20 aprile 2017
  - Soluzione primo esonero 2017
- Compiti 2018 ☐
- Soluzione compito 5 febbraio 2018 ☐
- Soluzione compito 19 febbraio 2018 ☐
- Testo e soluzione primo esonero 2018 ☐
- Soluzione compito 21 settembre 2018 ☐
- Soluzione compito 22 febbraio 2019 ☐
- Testo e soluzione esonero 18 aprile 2019 ☐
- Testo e soluzione esonero 7 giugno 2019 ☐
- Testo e soluzione compito 28 giugno 2019 ☐
- Testo e soluzione compito 24 luglio 2019 ☐
- Testo e soluzione compito 3 settembre 2019 ☐
- Testo compito 17 settembre 2019 ☐
- Testo e soluzione compito 12 novembre 2019 ☐

Argomento 3

Risultati esonero 18 aprile 2019

- Risultati esonero 18 aprile 2019

Risultati esonero 07 giugno 2019

- Risultati esonero 7 giugno 2019

Lista esonerati 2019

- Lista esonerati 2019

☐

Lista esonerati 2015

- Lista esonerati 2015

Risultati prova scritta 14 gennaio 2020

- Risultati prova scritta 28 giugno 2019
- Risultati prova scritta 24 luglio 2019
- Risultati prova scritta 3 settembre 2019
- Risultati prova scritta 17 settembre 2019
- Risultati prova scritta 12 novembre 2019
- Risultati prova scritta 14 gennaio 2020

D'ora in poi le registrazioni agli esami andranno effettuate presso il sito del SOL - Servizi di Segreteria On Line

La correzione del compito si terrà lunedì 24 febbraio ore 15.30 in aula verde

Argomento 4

Argomento 5

Argomento 6

Argomento 7

Argomento 8

Argomento 9

Argomento 10

Argomento 11

Argomento 12

Argomento 13

Argomento 14

Argomento 15

Argomento 17

Argomento 18

Argomento 19

Argomento 20

Argomento 21

Argomento 22

Argomento 23

(c) Copyright KITLab - Knowledge and Information Technologies Lab - Dipartimento di Matematica e Informatica