INDICAZIONI PER IL RECUPERO/RIPASSO/STUDIO ESTIVO
Classe 3AIIN 2020 - 2021

1. Ripassare tutti gli argomenti svolti a lezione. Ti allego copia del registro elettronico che riporta gli argomenti svolti. (Riferimenti: libro di testo, appunti e slide).

2. Rifare gli esercizi svolti in classe, in laboratorio e alcuni fra quelli indicati sul libro di testo.

3. Si raccomanda, per ciascun esercizio, di svolgere i seguenti punti:
   - individuazione dei dati di input e di output
   - individuazione delle strutture dati (disegno, dichiarazione in C/C++ e breve spiegazione del loro utilizzo)
   - applicazione della metodologia Top-Down: scomposizione del problema in sottoproblemi. Individuazione dei sottopgm: scrittura dei prototipi (indicare la direzionalità dei parametri). Di ciascun sottopgm deve essere indicata brevemente la funzione svolta.
   - scrittura del main pgm
   - descrizione (ad alto livello) di ciascun sottopgm utilizzando come linguaggio di progetto lo pseudocodice
   - codifica dei sottopgm in C/C++
   - elaborazione (scritta) di un piano di test
   - trace del pgm dei i casi previsti dal piano di test

4. Nella progettazione di un pgm si raccomanda di rispettare alcune regole per la scrittura di pgm facilmente leggibili e manutenibili:
   - scegliere opportunamente i nomi degli identificatori (costanti, tipi, variabili e sottopgm devono richiamare il loro uso o la funzione svolta)
   - indentare con cura
   - fare un uso appropriato dei commenti (sintetici e significativi)
   - utilizzare le costanti
   - scrivere una sola istruzione per riga
   - fare uso dei sottopgm e del passaggio di parametri
   - ogni sottopgm deve svolgere una sola funzione logica
   - non usare variabili globali (tranne dove viene esplicitamente consentito)
   - all'interno dei sottopgm non effettuare operazioni di I/O (tranne sottopgm di inserimento o stampa e casi eccezionali): utilizza i parametri o i valori di ritorno delle funzioni.

5. Rifare tutti gli quesiti/esercizi assegnati nelle prove scritte e nelle prove individuali di laboratorio.

6. Rispondere oralmente alle domande di verifica poste alla fine di ciascun capitolo del libro di testo.

7. Carica in Moodle i compiti svolti indicando chiaramente nel nome del file il numero di esercizio corrispondente (e anche la pag. se ti riferisci al libro di testo, da cui puoi scegliere altri esercizi).

8. Prima di fare i compiti assegnati ti conviene rivedere prima la teoria e successivamente gli esercizi svolti in classe e/o a casa.

9. Mi raccomando il linguaggio di progetto, il piano di test, la tabella di trace e il disegno dello stack/heap.

Per i linguaggi C/C++: **http://www.cplusplus.com/**
Per i linguaggi relativi al web: **https://www.w3schools.com/**

Esercizio 0 Algoritmi – Strutture di controllo - Programmi

Scrivere un algoritmo (utilizza il linguaggio di progetto) che risolva il seguente problema: dati due numeri N > 10 e K > 0 inseriti dall'utente, si stampi quanti sono i numeri interi compresi tra 10 e N incluso, tali che la somma delle proprie cifre sia uguale a K e sia la prima che l'ultima cifra del numero siano dispari.
Ad esempio con N = 100 e K = 8 la risposta sarebbe 4, poiché i numeri che soddisfano le condizioni elencate sono il 17, il 35, il 53 e il 71.

Esercizio n. 1 Procedure – Strutture di controllo

Scrivere una procedura che presi in input un carattere c e un intero dispari n (letti nel main()), stampa una forma, simile a una clessidra, come si deduce dai seguenti esempi (c = '*'):

```
n = 1     n = 3     n = 5      n = 7
  *        ***      *****     *******
            *        ***       *****
           ***        *         ***
                     ***         *
                    *****       ***
                                *****
                               *******
```

Esercizio n. 2 Funzioni – Vettori - Stack

Scrivi un pgm C++ che legge tre interi positivi n, h e k, con h < k, genera un vettore v di n (<= DIM_MAX) numeri interi casuali compresi tra h e k (estremi inclusi); stampa il vettore v.
Il pgm successivamente costruisce e stampa un altro vettore w di n elementi, i cui elementi w_i siano dati dal numero di elementi di v multipli di v_i.

Esempio Supponiamo che i due interi letti in input siano $n = 12$, $h = 2$ e $k = 18$. Sia inoltre $V = (10, 6, 5, 2, 7, 9, 3, 5, 15, 18, 17, 18)$ il vettore di $n = 12$ numeri interi casuali compresi tra $h = 2$ e $k = 18$. Allora il vettore W sarà il seguente: $W = (1, 3, 4, 5, 1, 3, 6, 4, 1, 2, 1, 2)$.

Nota: Utilizza i sottopgm.

Disegna lo stack.

Esercizio n. 3 Funzioni – Stringhe - Stack

Scrivi un pgm C++ che legge in input una stringa str composta da singole cifre numeriche alternate con i simboli "+" e "-". Calcolare e stampare il valore dell'espressione aritmetica rappresentata da str.

NOTA: il Calcolo deve essere svolto da un sottopgm. E' una funzione?

Esempio
Sia str = "3 - 5 + 4 + 7 - 2". Il valore dell'espressione rappresentato da str è 7.
Sia str = "- 3 + 8 - 7 - 2". Il valore dell'espressione rappresentato da S è -4.

Disegna lo stack.

Esercizio n. 4 Funzioni – Strutture di controllo

```
int main(){
int i = 3;
char x[] = "P", y[] = "LO";

        i++;
        printf("%c", *x);
        i = f(x, y, i-1);

        return 0;
}

int f(char * x, char * y, int k) {
        if (k)
                k = g(k, x, y);

        return k+1;
}

int g(int i, char * a, char * b) {
int j ;

        if(i % 2 == 0)
                j = 0;
        else
                j = 1;

        printf("%c", b[j]);
        i = f(a, b, i-1);
        return i;
}
```

Disegna stack e determina output.

Esercizio n. 5 Array – Record – Funzioni – Ricerca – Sort - Stringhe

Scrivere un pgm C++ che gestisce un array di NMAX elementi; ciascun elemento è costituito da una stringa S di DIMSTR caratteri e da un intero C.
L'array è inizialmente vuoto e la gestione avviene accettando da input una serie di comandi. I comandi sono:

I - Inserimento: accettazione da input di una stringa NS; se non esiste alcun elemento con S = NS si inserisce un nuovo elemento nell'array in cui si memorizza NS in S e si pone C ad uno; in caso contrario si incrementa C di una unità nell'elemento NS = S;

C - Cancellazione: accettazione da input di una stringa NS; se esiste un elemento con S = NS si decrementa C di una unità; se C raggiunge il valore zero si elimina l'elemento dall'array attraverso un'operazione di shift degli elementi verso sinistra; se l'elemento non esiste si visualizza un messaggio d'errore;

V - Visualizzazione di tutti di elementi presenti nell'array in ordine decrescente di occorrenze;

F - Fine lavoro.

Rilevare le eventuali condizioni di errore e visualizzare un messaggio opportuno.

Sottogm richiesti (**con parametri**):

- menu(): presenta sul video il menu dei comandi permessi;
- inserimento(): inserisce una stringa nella struttura (v. algoritmo sopra);
- cancellazione(): cancella una stringa dalla struttura (v. algoritmo sopra);
- cercaStr(): ricerca nella struttura la stringa da inserire o da cancellare;
- visualizzazione(): visualizza il contenuto della struttura (v. modalità sopra);
- sort(): ordina tutti gli elementi presenti nella struttura in ordine decrescente di occorrenze.

Esercizio n. 6 Record – Vettori – Stringhe - Funzioni

Dati due vettori di record contenenti alcune informazioni su quadri e pittori, aventi rispettivamente le seguenti strutture:

typedef struct{ **char titolo[DIM_STR_TITOLO];** **char autore [DIM__STR_PITTORE];** **double altezza;** **double larghezza;** **}quadro;**	**typedef struct{** **char nomePittore[DIM_STR_PITTORE];** **char genere[DIM_STR_GENERE];** **}pittore;** L'attributo nomePittore (= autore) è chiave primaria. L'attributo genere è una chiave secondaria.

Esempio:

Titolo	Autore	Altezza	Larghezza
Tramonto	Rossi	1.20	2.40
Tramonto	Verdi	1.20	2.40
Egitto	Verdi	0.60	0.60
Andrea	Bianchi	0.60	0.40

NomePittore	Genere
Rossi	paesaggista
Bianchi	ritrattista
Verdi	paesaggista

Scrivere le seguenti funzioni C++:

a. Stampare le informazioni di tutti i quadri di forma quadrata;

Nel nostro esempio:

Egitto	Verdi	0.60	0.60

b. Determinare il numero di quadri fatti da un certo pittore (il nome del pittore è un parametro e il numero di quadri non deve essere stampato dalla funzione);

Nel nostro esempio: input (Verdi) output (2)

c. Stampare per ogni pittore il numero dei suoi quadri;

Nel nostro esempio:

Rossi 1
Bianchi 1
Verdi 2

Scrivere inoltre un main di prova (richiama in sequenza le funzioni con parametri):

- leggiPittori() [richiedere e controllare il numero di pittori, numPittori ≤ MAX_PITTORI];
- leggiQuadri() [richiedere e controllare il numero di quadri, numQuadri ≤ MAX_QUADRI];
- funzioneA();
- funzioneB();
- funzioneC().

Ti consiglio, come test, di stampare il contenuto dei due vettori di record dopo che hai effettuato l'inserimento.
Disegna lo stack.

Esercizio n. 7 Funzioni - Ricorsione - Stack

Si scriva una funzione ricorsiva C/C++ con la seguente interfaccia

int stessoSegno(const int a[], int n);

che considera la porzione dell'array a compresa fra l'indice 0 e l'indice n e verifica se gli elementi di tale porzione di array sono tutti dello stesso segno (o tutti positivi o tutti negativi). La funzione fornisce

- 1 se gli elementi sono tutti dello stesso segno
- 0 altrimenti.

Ad es., se a={-1,-3,-1,4,-5}

- **stessoSegno**(a,2) = 1
- **stessoSegno**(a,4) = 0

Si mostri poi il funzionamento del seguente programma, che invoca la funzione definita precedentemente, utilizzando i record di attivazione (disegno dello stack).

```
int d(int*y, int m,int a[]){
int j;
	for(j=*y; j<m; j++){
		a[j] = a[j] - *y;
		(*y)=a[j]*m;
	}

	if(stessoSegno(a,m))
		m++;
	else
		m--;
	return m;
}

int main(){
int n=2, a[4]={-1,0,1,-2}, x=0;

	x = d(&x,n+1,a);

	return 0;
}
```

Disegna lo stack.

Esercizio n. 8 Record – Array – Tipi enumerativi – Funzioni – File di testo

Un programma che gestisce annunci può essere definito a partire dalle seguenti costanti e dai seguenti tipi:

```
const int MAX_STR = 100;
const int MAX_ANNUNCI = 200;
enum tipoCategoria {HOBBY, AUTO, CASA, SPORT};

struct tipoData {
	int giorno;
	int mese;
	int anno;
};

struct tipoAnnuncio {
	tipoCategoria categoria;
	tipoData dataPubblicazione;
	float prezzo;
	char descrizione[MAX_STR];
};

struct archivioAnnunci {
	tipoAnnuncio annunci[MAX_ANNNUNCI];
	int numAnnunci;
};
```

- tipoAnnuncio definisce il contenuto di un annuncio: categoria, data di pubblicazione, prezzo di vendita e descrizione testuale.
- le categorie possibili sono quelle indicate dal tipo tipoCategoria, mentre le date sono rappresentate attraverso il tipo tipoData.
- la struttura archivioAnnunci memorizza gli annunci nell'array annunci; il numero di annunci presenti è indicato da numAnnunci.

Realizzare le seguenti funzioni:

float prezzoMedio(const archivioAnnunci* a, tipoCategoria c) restituisce il prezzo medio degli annunci di categoria c.

bool piuNumerosa(const archivioAnnunci* a, tipoCategoria* puntCategoria) trova la categoria più numerosa e la restituisce al chiamante mediante puntCategoria. La funzione restituisce *true* se c'è almeno un annuncio, *false* altrimenti.

void stampa(const archivioAnnunci* a) stampa a video il contenuto dell'archivio. Ogni annuncio viene stampato su una riga separata che comincia con il carattere +, =, o - se l'annuncio ha un prezzo rispettivamente maggiore, uguale, o minore del prezzo medio degli annunci della sua stessa categoria. Seguono il prezzo dell'annuncio e la sua descrizione testuale.

Esempio:

+ 120.5 Barca radiocomandata
- 23 Aquilone
= 350 Credenza
- 450 Snowboard
- 650 Tavola surf
+ 3500 Paracadute

int seleziona(const archivioAnnunci* a, const char *stringaCercata, tipoAnnuncio *v) riempie l'array v (allocato staticamente, non abbiamo ancora visto l'allocazione dinamica nello heap) con tutti gli annunci il cui campo descrizione contiene la stringa indicata da stringaCercata. La funzione restituisce il numero di annunci inseriti in v.

bool salva(const archivioAnnunci* a, const char nomeFile[]) salva nel file indicato da nomeFile il contenuto dell'archivio ordinando gli annunci per data crescente. Rispettare il formato indicato dal seguente esempio:

12/3/2014 hobby 120.5 Barca radiocomandata
5/5/2014 sport 3500 Paracadute
23/8/2014 casa 350 Credenza
11/11/2014 hobby 23 Aquilone
8/1/2015 sport 450 Snowboard
23/1/2015 sport 650 Tavola surf

Esercizio n. 9 File di testo - Matrici

Sia dato un file di testo che contiene la descrizione di un insieme di rettangoli posizionati su una matrice di punti. Sia definito sulla matrice un sistema di coordinate cartesiane dove ciascuna coordinata è un numero intero che varia da 0 a N estremi inclusi, l'origine è il punto in alto a sinistra e le coordinate aumentano da sinistra a destra e dall'alto al basso. N è una costante nota a priori e definita attraverso una direttiva define.
Ogni riga del file contiene la descrizione di un rettangolo nel seguente formato:

<x1> <y1> <x2> <y2>

dove (<x1>, <y1>) sono le coordinate del punto in alto a sinistra del rettangolo mentre (<x2>, <y2>) sono le coordinate del punto in basso a destra (quindi x1 x2 e y1 y2). Il numero totale di rettangoli (ovvero di righe del file) non è noto a priori. Non assumere alcun tipo di ordinamento delle righe stesse.

Scrivere un programma in C/C++ per verificare che tutti i rettangoli descritti nel file siano privi di sovrapposizioni (occupino cioè caselle della griglia disgiunte).

Il programma deve controllare il file e, dopo una opportuna elaborazione, scrivere il messaggio "Nessuna sovrapposizione tra i rettangoli" se il file soddisfa il criterio oppure "I rettangoli hanno (almeno) una sovrapposizione" in caso contrario.

Ad esempio (con N pari a 10), un file con la seguente descrizione di rettangoli:

2 1 6 2
8 7 10 9
5 2 9 6
0 8 5 8

produrrà il messaggio "I rettangoli hanno (almeno) una sovrapposizione".

Ad esempio (con N pari a 10), un file con la seguente descrizione di rettangoli:

2 1 6 2
8 7 10 9
5 2 9 6
0 8 5 8

corrisponde alla disposizione dei rettangoli mostrata in figura, dove il primo ed il terzo rettangolo si sovrappongono (nella visualizzazione si ha il numero di occupazioni).

```
_ _ _ _ _ _ _ _ _ _ _
_ _ 1 1 1 1 1 _ _ _ _
_ _ 1 1 1 2 2 1 1 1 _
_ _ _ _ _ 1 1 1 1 1 _
_ _ _ _ _ 1 1 1 1 1 _
_ _ _ _ _ 1 1 1 1 1 _
_ _ _ _ _ 1 1 1 1 1 _
_ _ _ _ _ _ _ _ 1 1 1
1 1 1 1 1 1 _ _ 1 1 1
_ _ _ _ _ _ _ _ 1 1 1
_ _ _ _ _ _ _ _ _ _ _
```

Esercizio n. 10 Funzioni – Array di stringhe – Matrici - Stringhe

Un commesso viaggiatore deve recarsi in N città. Iniziando da una di esse e conoscendo le varie distanze tra le città egli vuole completare il giro compiendo il tragitto più corto possibile e tornando al punto di partenza.
A tal fine, adotta la seguente strategia: scegliere come tappa seguente, a partire da ciascuna città, la città più vicina tra quelle non ancora visitate.
Si suppone di disporre dei seguenti dati: l'elenco delle città e le rispettive distanze reciproche (in chilometri).
Scrivere un programma che, inserito il nome della città di partenza, indichi il percorso del viaggiatore conforme alla precedente strategia con l'indicazione del totale dei chilometri da percorrere.

Esercizio n. 11 Funzioni – Vettori - Record – Sort - Ricerca

Si vogliono definire le strutture opportune che permettano di rappresentare un appuntamento per la vaccinazione Covid.
Un appuntamento è definito dalla data dell'appuntamento e dal tipo di vaccino utilizzato (Moderna, Pfitzer, AstraZeneca, Janssen)

Dopo aver definito le opportune strutture dati scrivere:

- una funzione *ultimoGiornoPossibile()* che, dati come parametri un vettore di appuntamenti e la sua lunghezza (numAppuntamenti), un tipo di vaccino e la quantità di vaccini disponibili per quel tipo, ritorni la data in cui verrà fatta l'ultima iniezione di quel vaccino, considerando che gli appuntamenti all'interno del vettore sono in ordine di data crescente e che le iniezioni verranno fatte nell'ordine indicato negli appuntamenti.

Ora supponi poi che il vettore non sia ordinato:

a) insertAppuntamenti(); //insert di massa leggendo i dati (non necessariamente ordinati per data) da file di testo
b) stampaAppuntamenti();
c) shuffle(); //mescola, casualmente, l'ordine degli appuntamenti
d) sort(); //ordina gli appuntamenti per data crescente.

La procedura di sort utilizza una funzione dataMinoreUguale() che confronta due date.

Esercizio n. 12 Record – Vettori – Funzioni - Sort

Riempire una sequenza (mazzo) di 52 carte da gioco con le carte di un mazzo francese. Successivamente riempire una seconda sequenza (mano) con 13 carte casualmente estratte dal mazzo; visualizzare la mano di carte ordinata secondo il valore delle carte. Aggiungere carte casualmente estratte dal mazzo alla mano di carte fintantoché nella mano non si ha un tris di carte, visualizzando le carte estratte. Infine visualizzare la nuova mano di carte ordinata seconda il valore delle carte. Per svolgere questo compito di definisca ed utilizzi 5 funzioni con le seguenti finalità.

- Visualizzare una data carta da gioco
- Visualizzare una data sequenza di n carte da gioco
- Ordinare una data sequenza di n carte da gioco secondo il valore delle carte
- Produrre una carta da gioco casualmente selezionata in una data sequenza di carte da gioco formata da un numero di carte memorizzato in una variabile puntata da un dato puntatore n a variabile intera; rimuovere dalla sequenza la carta selezionata; decrementare di una unità il numero di carte puntato da n.
- Produrre 1 se una data sequenza di n carte da gioco contiene un tris; produrre 0 altrimenti.

Esercizio n. 13 **Funzioni – Puntatori – Stack – Allocazione dinamica**

Disegnare lo stack e determina l'output:

```
double x = 12.3;  //var globale. A scuola è vietato usarle!

int main(){
char z[] = "puff";
char* t;

	int x = z[0] ? 2 : 0;
	int* p = &x;
	t = zip((*p)+2, z, &x);
	printf("Effetto esercizio: %d %s\n",*p,t);

	return 0;
}

char * zip(int x, char *s, int *m){
int i,k;

	char* t = (char*)malloc((x+1)*sizeof(char));
	t[x] = '\0';
	for (i = --x; i > 0; --i){
		k = (i + *m) % x;
		*(t+k) = s[i];
	}
	t[x] = 'a';
	return t;
}
```

Esercizio n. 14 Funzioni – Allocazione dinamica - Stack

Dati due array di interi **A** e **B** di numElem elementi, il loro *shuffle* è l'array di interi **C** di 2*numElem elementi ove il primo elemento (quello di posto **0**) è il primo elemento di **A**, il secondo è il primo elemento di **B**, il terzo è il secondo elemento di **A**, il quarto è il secondo elemento di **B**, e così via.
Scrivete la funzione **shuffle()** che abbia come parametri due array di interi e il loro numero di elementi e restituisca lo *shuffle* dei due array passati per argomento. Il vettore che deve contenere lo shuffle dei due array deve essere allocato dinamicamente.

Disegna lo stack e lo heap.

Esercizio n. 15

Scrivi un pgm che ti permetta di giocare a mastermind.

Mastermind è un gioco crittoanalisi per due giocatori, in cui un giocatore (tu, nel nostro caso), il "decodificatore", deve indovinare il codice segreto composto dal suo avversario (il computer, nel nostro caso), detto "codificatore".

Se non conosci il gioco, cerca le regole in Internet.

- Utilizza la modalità testo (non grafica).
- Scomponi il problema in sottoproblemi.
- Utilizza il passaggio di parametri.

Esercizio n. 16 HTML + CSS

Organizza il lavoro svolto in una cartella come ti è stato insegnato.

1. Relativamente al linguaggio **HTML(5) + CSS(3)** leggi e studia i due capitoli del libro di testo (+ appunti/slide/esempi visti in lab.).
Progetta e realizza un semplice sito (tema scelto a piacere) con almeno 5 pagine applicando tutti (o quasi) i concetti descritti nei capitoli (NON usare JavaScript, PHP o altri linguaggi non trattati a lezione).

Per poter sviluppare un sito web in modo che sia utilizzabile e facilmente gestibile dobbiamo porci delle domande prima di iniziare, dobbiamo cioè realizzare l'**analisi** del nostro progetto.

1. Qual è il target di utenza che vogliamo raggiungere?
2. Che cosa vogliamo pubblicare/comunicare?
3. Come organizziamo il contenuto (struttura del sito)?
4. Come lo pubblichiamo?
5. Come verifichiamo la sua completezza e correttezza?
6. Come lo teniamo aggiornato?
7. Come lo pubblicizziamo?

Voglio la risposta (scritta) a queste domande in una relazione che descrive l'analisi del progetto del sito.

La spiegazione delle 7 fasi la puoi trovare sul libro di seconda della disciplina Scienze e tecnologie applicate (oppure cerca in internet).

Ti ricordo il sito: