Prova scritta Programmazione Procedurale con Lab. - 18 Giugno 2024

Nome e Cognome: _____ Matricola: _____

1. [5 punti] A cosa servono i file di header (file *.h*). Cosa viene di solito inserito in questi file? Cosa accade se si inserisce una definizione di variabile globale *int a* in un file .h che viene incluso in due different file? Rispondere ai tre quesiti.

2. [6 punti] Definire una funzione *rotate90()* che prenda in input una matrice di dimensione $n \times n$, e ne crei una $n \times n$ che corrisponde ad una rotazione in senso antiorario di 90 gradi, come in esempio. Inoltre, la funzione stampa la matrice risultato.

3. [6 punti] Data la seguente definizione di *struct Node*, definire su foglio protocollo una funzione di nome *unisci_alterna()* che riceve come parametri due liste e ne crea una nuova inserendo i valori delle due liste in modo alternato. Per esempio, se le due liste passate sono 3-1-12 e 63-115-7, la lista creata e ritornata come risultato sarà 3-63-1-115-12-7. Nota: *i)* supporre le due liste di lunghezza uguale *ii)* definire la funzione senza calcolare la lunghezza delle liste.

```
1 struct Node {
2   int info;
3   struct Node* pNext;
4 };
```

4. [6 punti] Dire quali compilazioni provocano errore a causa del linker (e perché): 1) *gcc –c main.c*, 2) *gcc –o write write.c*, 3) *gcc write.c main.c -o main*, 4) *gcc –o main main.c*. In caso il punto 3) ritorni un errore, descrivere come può essere corretto. Infine, che tipo di *linkage* hanno *count*, *i*, e *mywrite*? Cosa stampa il programma *output*? Elencare tutte le definizioni e dichiarazioni in ogni file.

main.c
```
int count;
int count= 3;
void mywrite(int count);
int main(void) {
extern int count;
do {
   mywrite(count);
}while(count >=0);
}
```

write.c
```
#include <stdio.h>
int i= 1;
static int count= 4;

void mywrite(int a) {
   a--;
   printf("%d\n", count= count - i);
}
```

2) Errore, non esiste file main da compilare insieme a file main.c
4) Manca la definizione di mywrite

1) non ritorna nessun errore
In main, count viene prima dichiarato (tentativo definizione) e sotto definito, myWrite viene dichiarata, main viene definita, count viene dichiarato dentro main
In write.c i viene definita, count viene definita, myWrite viene definita

Viene stampato 3
2
1
0
-1
-2 e così via

5. [7 punti] Cerchiare le affermazioni vere dato *int a[5]= {129, INT_MIN, INT_MIN | INT_MAX, 262142, 262168};* short int *p = (short*) a; char *q= (char*) a; *q[1]= 1;* sapendo che i tre tipi usati occupano 4, 2, e 1 byte, e $262144 = 2^{18}$ (valori rappresentati in *little endian* e complemento a due). Scrivere la mappa di memoria e giustificare le affermazioni (vere o false). A. $\&a[5] - (a+2) - q[1] - 2$ B. $^*(p+2)\,?\,p[7]\ :\ !(p[4]+p[5]+p[7])$ C. $(((int)(p+10) - (int)(a+1)) + q[16] - q[18])\,\%\,6$

Esercizio 1

- I file .h (header) nel linguaggio C sono utilizzati per dichiarare funzioni, variabili, definire tipi di dati e macro che possono essere utilizzati in più file sorgente (.c). Questi file svolgono un ruolo cruciale nella modularità e nella gestione del codice, permettendo di organizzare e riutilizzare il codice in modo più efficiente. Possono
essere inclusi utilizzando la direttiva del preprocessore #include

- Dichiarazioni di Funzioni,
Definizioni di Macro: esempio #define COSTANTE 10,
Definizioni di Tipi: esempio
typedef struct {
    int campo1;
    float campo2;
} EsempioStruttura;
Dichiarazioni di Variabili Esterne: esempio extern int variabileGlobale;
Inclusione di Altri Header: esempio #include "altroHeader.h"

- Se inserisci una definizione di variabile in un file .h e lo includi in due file sorgente diversi, otterrai un errore di compilazione a causa di una definizione multipla della stessa variabile. Questo accade perché ogni volta che un file header viene incluso in un file sorgente, il contenuto del file header viene letteralmente copiato nel file sorgente durante il preprocessing.

Esercizio 2
https://medium.com/enjoy-algorithm/rotate-a-matrix-by-90-degrees-in-an-anticlockwise-direction-6326e80bb211

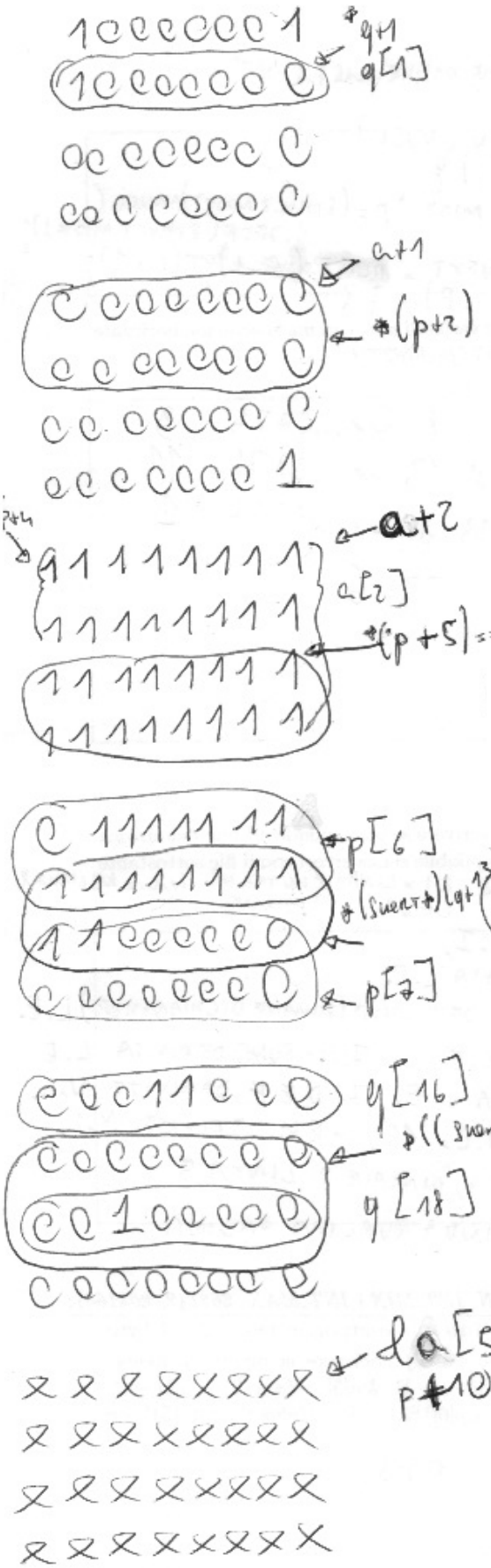
```
void rotate90(int m[][], int n)
{
   int temp[n][n];
   for(int i = 0; i < n; i = i + 1)
   {
      for(int j = 0; j < n; j = j + 1)
      {
         temp[n-j-1][i] = m[i][j];
      }
  }
  // stampa di m
  return;
}
```

Esercizio 3

```
struct Node* unisci_alterna(struct Node* l1, struct Node l2) {
  if (l1 == NULL)
    return NULL;
  struct Node* temp = l1;
  struct Node* lista_finale = NULL;
  struct Node* ultimo = NULL;
  while (l1 != NULL) {
    struct Node* new1 = (struct Node*) malloc(sizeof(struct Node));
    new1 -> info = l1 ->info;
    struct Node* new2 = (struct Node*) malloc(sizeof(struct Node));
    new2 -> info = l2 ->info;
    new2 -> pNext = NULL;
    new1 -> pNext = new2;

    if (lista_finale == NULL) {
      lista_finale = new1;
    else
      ultimo -> pNext = new1;

  ultimo = new2;
  temp = temp -> pNext;
  }
  return lista_finale
}
```



A) $\underbrace{la[5] - (a+2)}_{3} - q[1] - 2 == 0$

$3 - 1$

FALSA

C) $\underbrace{(INT)(p+10) - (INT)(a+1)}_{16} + q[16] - q[18]$

$+ 24 - 4$

$== 36 \% 6 = 0$

FALSA

B) $*(p+2) == 0$

QUINDI

$!(p[4] + p[5] + p[7])$

$!(-1 + (-1) + 3) == !1 == 0$

FALSA