Α.Μ.:

ΟΝΟΜΑΤΕΠΩΝΥΜΟ:

Θέμα 1 [4 μονάδες]

Γράψτε μια συνάρτηση με όνομα fun() που να δέχεται ως παραμέτρους έναν πίνακα ακεραίων a, το πλήθος των στοιχείων του n, μια θέση στον πίνακα pos και να επιστρέφει τον πίνακα a με όλα τα στοιχεία μετά τη θέση pos να έχουν μετακινηθεί μια θέση προς τα αριστερά ενώ στην πλέον δεξιά θέση του πίνακα να έχει τοποθετηθεί η τιμή μηδέν. Για παράδειγμα αν η συνάρτηση κληθεί για a={3,7,6,8,9}, n=5, pos=2 τότε να επιστρέφει a={3,7,8,9,0}. Αν το pos έχει τιμή εκτός των ορίων [0,n-1] η συνάρτηση να επιστρέφει χωρίς να αλλάζει τον πίνακα. Γράψτε πρόγραμμα που να καλεί τη συνάρτηση fun από την main για τις τιμές που αναφέρθηκαν.

```
#include <iostream>

using namespace std;

void fun(int* a, int n, int pos){
	for(int i=pos;i<n-1;i++)
		a[i] = a[i+1];
	a[n-1]=0;
}

int main(){
	int a[]={3,7,6,8,9};
	fun(a,5,2);
	for(int i=0;i<5;i++)
		cout << a[i] << " ";
	cout << endl;
}
```

Θέμα 2 [4 μονάδες]

Να δηλωθεί ο κόμβος μιας απλά συνδεδεμένης λίστας που να περιέχει ως δεδομένα ένα ζεύγος ακεραίων τιμών. Δημιουργήστε μια απλά συνδεδεμένη λίστα με 3 κόμβους. Γράψτε μια συνάρτηση με όνομα swap_all που να αντιμεταθέτει το ζεύγος ακεραίων κάθε κόμβου και για όλους τους κόμβους της λίστας. Καλέστε τη συνάρτηση από τη main() και εμφανίστε τη λίστα. (π.χ. η λίστα: (4,5)->(3,1)->(7,9)->NULL θα πρέπει να γίνει (5,4)->(1,3)->(9,7)->NULL).

```
#include <iostream>

using namespace std;

struct node {
	int a, b;
	struct node* next;
};


void swap_all(struct node* anode) {
	while (anode != NULL) {
```

```
            std::swap(anode->a, anode->b);
            anode = anode->next;
      }
}

int main() {
      struct node* node1 = new node {4, 5, NULL};
      struct node* node2 = new  node {3, 1, NULL};
      struct node* node3 = new  node {7, 9, NULL};
      node1->next = node2;
      node2->next = node3;

      cout << "(" << node1->a << " " << node1->b << ")";
      cout << "(" << node2->a << " " << node2->b << ")";
      cout << "(" << node3->a << " " << node3->b << ")" << endl;
      swap_all(node1);
      cout << "(" << node1->a << " " << node1->b << ")";
      cout << "(" << node2->a << " " << node2->b << ")";
      cout << "(" << node3->a << " " << node3->b << ")" << endl;
}
```

Θέμα 3 [2 μονάδες]

Ένας σωρός μεγίστων έχει αποθηκευτεί στον πίνακα: [22,13,10,8,7,6,2,4,3,5]. Σχεδιάστε το σωρό ως δένδρο. Στο ίδιο σχήμα προσθέστε την τιμή 25. Διαγράψτε τη μεγαλύτερη τιμή του **αρχικού σωρού** (δηλαδή χωρίς την προσθήκη της τιμής 25) και σχεδιάστε εκ νέου το σωρό ως δένδρο.