| Question No. 01 |
| --- |
| The following code finds the product of all elements in an array. Rewrite it using pointers.  int a[n];  int prod=1;  for (int i=0; i<n; i++) {  prod = prod \* a[i];  }  printf("%d", prod); |
| **int main(){**  **int n = 5;**  **int a[5] = {1, 2, 3, 4, 5};**  **int prod = 1;**  **for (int i = 0; i < n; i++)**  **{**  **prod = prod \* (\*a + i);**  **}**  **printf("%d", prod);**  **}** |

| Question No. 02 |
| --- |
| To dynamically allocate an array of n elements and int type, we use the following syntax.  int \*a = (int\*) malloc(n\*sizeof(int));  How to declare an array of double type and 5 elements. How much space does it take? |
| **double \*a = (double\*) malloc(n\*sizeof(double));**  **It would take 40 Bytes.** |

| Question No. 03 |
| --- |
| John declares an variable length array as follows.  int n;  scanf(“%d”, &n);  int a[n] = {1, 2, 3, 4, 5};  What do you think will happen if you try to input   1. 4 2. 5 3. 6 |
| We can not initialize the array like that because of variable size. |

| Question No. 04 |
| --- |
| Try to find the maximum length of a dynamically allocated array in your setup.  int n = 1000000000;  int \*a = (int\*) malloc(n\*sizeof(int));  In the above code, change the value of n, and see what values the allocation fails. Remember, if the allocation fails, you will get a null pointer. |
| **NO IDEA** |

| Question No. 05 |
| --- |
| Write a function that will take as input an array and return an array via parameter. The signature will be void makeCopy(int n, int input[], int output[] ). Here the output array will store your return value. |
| void makeCopy(int n, int a[], int b[]) {  for (int i = 0; i < n; i++)  b[i] = a[n - 1 - i];  } |

| Question No. 06 |
| --- |
| Do the same task, but return by pointer. The signature will be int\* makeCopy(int n, int input[]). Hint: Use dynamic allocation. |
|  |

| Question No. 07 |
| --- |
| We wrote a program to find all positions of a character in a string. Now write a program to find all positions of a pattern string in a text string. Use the strstr function.    Example: Suppose text = “pro programming product” and pattern = “pro”.  Your program should print:    pro found at position 0.  pro found at position 4.  pro found at position 16. |
|  |

| Question No. 08 |
| --- |
| Both bubble sort and selection sort use the swap function. Which one do you think uses it more? How can you track how many times swap is called? Hint: Use static variables in the swap function. |
|  |

| Question No. 09 |
| --- |
| Modify the bubble sort function so that it sorts the array in reverse sorted order, ie. from the largest to smallest. For example reverse sorting a = {3, 4, 2, 5, 1} should result in {5, 4, 3, 2, 1}.  Hint: You only need to change one character from the above code. |
|  |

| Question No. 10 |
| --- |
| Remember the Bubble Sort function.  void sort(int n, int a[]) {  for (int steps=0; steps<n; steps++) {  for (int i=0; i+1<n; i++) {  if (a[i] > a[i+1]) {  swap(&a[i], &a[i+1]);  }  }  if (is\_sorted(n, a)) break;  }  }  Implement the is\_sorted() function to optimize the function. |
|  |

| Question No. 11 |
| --- |
| Remember the Selection Sort function.  void sort(int n, int a[]) {  for (int steps=0; steps<n; steps++) {  int minelement = a[steps], pos = steps;  for (int i=steps; i<n; i++) {  if (a[i] < minelement) {  minelement = a[i];  pos = i;  }  }  swap(&a[steps], &a[pos]);  }  At every step we took the smallest element and moved it to the beginning. Rewrite the same function, but this time take the largest element and move it to the end, every iteration.  Example: Suppose you are sorting the array {2, 3, 1, 5, 4}. Then at the end of each step the array should look like as follows:  Step 1: 2, 3, 1, 4, 5  Step 2: 2, 3, 1, 4, 5  Step 3: 2, 1, 3, 4, 5  Step 4: 1, 2, 3, 4, 5  Step 5: 1, 2, 3, 4, 5 |
|  |

| Question No. 12 |
| --- |
| Write a function takeNegatives which takes the negative numbers from an array in, and stores them in another array out. You should return how many negative elements there are in the array. The signature should be.  int findNegatives(int n, int in[], int out[])  For example, if in = {1, -2, -3, 4, 5}. Then the function should return 2 and out should be {-2, -3}. |
|  |

| Question No. 13 |
| --- |
| Write a makeReverse function that takes an array and returns its reverse by pointer. To make sure you don’t accidentally reverse the array itself, we use the const modifier in the signature. The signature should be  int \* makeReverse(int n, const int a[]); |
|  |