1. Consider the following statement:

int Y[5]={4, 2, 3, 1, 7}, X[5];

 Which one of the following methods is the correct way to copy array Y to X?

1. X = Y;
2. X[] = Y[];
3. for (i=0 ; i<5; i++) X[i] = Y[i] ;
4. X[5] = Y[5];
5. Consider the following array: int x[]={2, 4, 6, 9, 13}; Using the linear search algorithm, how many comparisons will be conducted if the target value is **19**?
6. 0
7. 6
8. 5
9. 1

#define SIZE 3 //3 points

#include <stdio.h>

void fun1(int a, int \*x, int n){

 int i;

 for(i=0; i < n; i++)

 a = a+\*x;

}

int main(){

 int a[SIZE]={1,3,0},i;

 for(i=0;i<SIZE;i++)

 fun1(a[i],&a[2-i],3);

 for(i=0;i<SIZE;i++)

 printf("%d\n",a[i]);

}

**1**

**3**

**0**

#include<stdio.h>

int main(){

 int j=0,x =13,y[4];

 while(x != 0){

 y[j] = x%2;

 x /= 2;

 j++;

 }

 for(j--;j>=0;j--)

 printf("%d ",y[j]);

}

**1 1 0 1**

#include<stdio.h>

void test1(int \*x, int y){

 \*x = y + \*x;

 y = 2\*y;

}

void test2(int \*x, int y){

 x[3] = x[2];

 y = 3\*y;

}

int main(void){

 int x[5] = {4,2,5,1}, y = 8, i;

 test1(&y,x[1]);

 test1(&x[1], x[2]);

 test2(x, x[4]);

 printf("%d\n",y);

 for(i=0; i<5; i++)

 printf("%d ",x[i]);

 return 0;

}

10

4 7 5 5 0

1. If a Selection sort algorithm is applied to sort an array *x* in a decreasing order (i.e., from highest to lowest), what will be the new value of the array *x* with values {8, 4, 60, 1, 2, 25, 15} after the first pass of the selection sort algorithm?

A. 1 2 4 8 25 15 60

B. 60 4 8 1 2 25 15

C. 1 2 4 8 15 25 60

D. 60 8 4 1 2 25 15

1. Array elements are always stored in \_\_\_\_\_\_\_\_ memory locations.
2. Sequential
3. Random
4. Sequential and Random
5. None of the above
6. What will be printed after execution of the following code?

int main()

{

 int arr[10] = {1,2,3,4,5};

 printf("%d", arr[5]);

}

1. Garbage Value
2. 5
3. 6
4. 0
5. What is the output of the following code

 int A[] = {2, 0, 3, 4, 1};

 printf("%d %d", A[A[4]]+2, A[A[2]-1]);

1. 6 1
2. 2 3
3. 6 -1
4. None of the above is true
5. When an array is passed as an argument to a function, what gets passed is:
6. The address of the size of the array
7. The value of the first element of the array.
8. The address of the first element of the array.
9. The number of elements of the array.

#include <stdio.h>

#define SIZE 4

void test (int \*b, int a, int i);

int main() {

int y[SIZE],i;

for(i=1;i<SIZE;i++) {

 y[i]=i;

 test(&y[i-1],y[i],i);

}

for(i=0;i<SIZE;i++)

 printf("%d\t",y[i]);

return 0;

}

void test (int \*b, int a, int i) {

 \*b=a+i;

 a=a\*2; }

**2 4 6 3**

[5 points]

int x[] = {3, 4, 1, 2, 6, 5};

int k;

for(k = 2; k <= 4; k++){

x[k - 2] = x[k - 2] + x[k + 1];

x[k] = x[k - 1];

}

for(k = 0;k <= 4; k++)

printf("%d ", x[k]);

5 10 9 4 4

[5 points]

#include <stdio.h>

int funcn(int a, int b, int\* c, int\* d);

int main(void) {

int a = 4,y = 3, x[] = {2, 5};

printf("%d ",funcn(a, x[1], &x[0], &y));

printf("%d %d %d %d",a, x[0], x[1], y);

return 0;

}

int funcn(int a, int b, int\* c, int\* d) {

a = 3 \* b;

b = 5;

\*c = 2;

\*d = 4 + \*c ;

return a + b + \*d;

}

26 4 2 5 6

1. Which of the following statements will read and store 4 input values into an integer array *arr1* of size 4 in reverse order?
2. for (i = 0; i < 4; i++) scanf("%d",&arr1[i]);
3. for (i = 3; i <= 0; i--) scanf("%d",& arr1[i]);
4. for (i = 3; i >= 0; i--) scanf("%d",&arr1[i]);

for (i = 0; i <= 4; i++) scanf("%d",&arr1[4-i]);

1. If an array name is passed as an argument to a function, what actually gets passed?
	1. The value of the first element of the array
	2. The values of all the elements of the array
	3. The address of the last element of the array
	4. The address of the first element of the array
2. What is the prototype for a function that computes a multiplication of array elements A and B each of size n and puts the result back in array A ?
	1. void mult(int A[ ], int B[ ], int n);

|  |
| --- |
| * 1. void mult (int A[n], int B[n], int n);
	2. int mult (int A[n], int B[n], int n);
	3. void mult (int\* A[ ], int\* B[ ], int n);
 |

1. Consider the following array:

 int m[3] ={7,2};

What is the value of m[2]?

1. 2
2. 7
3. 0
4. NULL

#include <stdio.h> // 6 points

void pick(int a, int \*b) {

 if(a < \*b) {

 \*b=\*b+3;

 a=a-10;

 }

 else {

 \*b=\*b-3;

 a=a+10;

 }

}

int main() {

 int i;

 int a[]={2,4,8,10,20,100,25,15,7,3,1};

 for(i=0;i<10;i=i+2) {

 pick(a[i],&a[i+2]);

 }

 for(i=0;i<10;i=i+2)

 printf("%d ",a[i]);

return 0;

}

2 11 23 28 4

1. An array elements are always stored in \_\_\_\_\_\_\_\_\_\_\_\_ memory locations.
2. Sequential
3. Random
4. Sequential and Random
5. None of the above
6. What will happen if in a C program if you assign a value to an array element whose subscript (or index) exceeds the size of the array?
7. The element will be set to 0.
8. The compiler would report an error.
9. The program may crash if some important data gets overwritten.

The array size would appropriately grow

1. The prototype of a function that takes two arrays of type **double** as input arguments and returns an array of type **double** (All of size **s** which is a variable) can be written as:
2. double fun(double a[ ],double b[ ],double \*c[ ], int s);
3. void fun (double a[s],double b[s],double \*c[s]);
4. void fun (double a[s],double b[s],double c[s]);
5. void fun(double a[ ],double b[ ],double c[ ], int s);

 int num[5] = {1, 2, 20, 2, 30};

 int x, y, z;

 x = ++num[1];

 y = num[1];

 z = num[++x];

 printf("%d, %d, %d\n", x, y, z);

**4, 3, 30**

 int num[5], k;

for(k = 0; k <= 4; k++){

 if( k % 2 == 0)

 num[k] = k + 2;

 else{

 num[k - 1] = k + 3;

 num[k] = num[k - 1] + 4;

 }

 }

 for(k = 4; k >= 0; k--)

 printf("%d ", num[k]);

**6 10 6 8 4**

#include <stdio.h>

int funcn(int a, int\* b);

int main(void) {

 int x[] = {1, 2, 3, 4};

 printf("%d %d\n",funcn(x[1], &x[3]), x[3]);

 printf("%d %d\n", x[1], x[3]);

 return 0;

}

int funcn(int a, int\* b) {

 \*b = 3 \* \*b;

 a = 2;

 return a + \*b;

}

**14 4**

**2 12**

#include <stdio.h>

int funcn(int a, int b, int\* c, int\* d);

int main(void) {

 int x = 3,y = 2, z[] = {5, 6};

 printf("%d\n",funcn(x, z[0], &z[1], &y));

 printf("%d %d %d %d",x, z[0], z[1], y);

 return 0;

}

int funcn(int a, int b, int\* c, int\* d) {

 a = 2 \* b;

 b = 6;

 \*c = 3;

 \*d = 5 + \*c ;

 return a + b + \*d;

}

24

3 5 3 8

1. Consider the following array: int x[]={2, 4, 6, 7, 9, 13};

Using the linear search function, how many comparisons will be conducted if the target value is 7?

* 1. 3
	2. **4**
	3. 6

**#include <stdio.h>**

**int main (void)**

**{**

 **int k, i, x = 14, bin[5];**

 **for (k = 0; x != 0; k++)**

 **{**

 **bin[k] = x % 2;**

 **x /= 2;**

 **}**

 **for (i = k-1; i >= 0; i--)**

 **printf("%d", bin[i]);**

 **return (0);**

**}**

**1110**

7. Consider the following array: int x[]={2, 4, 6, 7, 9, 13};

 Using the linear search function, how many comparisons will be conducted if the target value is 24?

1. 0
2. 1
3. 5
4. 7
5. **6 E**
6. Given the following function prototype:

**void getAverageAndSum(double a[], double b[],int size, double \*p, double \*e);**

Which of the function calls below is correct for the above function prototype? Assume **x** and **y** are 1-D arrays of type double. Assume **SIZE** is integer, and **sum** and **average** are variables of type double.

**A. getAverageAndSum( x, y, SIZE, &sum, &average); A**

B. **getAverageAndSum( x[SIZE], y[SIZE], SIZE, &sum, &average);**

C. **getAverageAndSum( &x, &y, SIZE, &sum, &average);**

D. **getAverageAndSum( x, y, SIZE, sum, average) ;**

E. None of the above is correct.

**int x[5], k;**

**for(k = 0; k <= 4; k++){**

 **if( k % 2 == 0)**

 **x[k] = k + 2;**

 **else{**

 **x[k – 1] = k + 3;**

 **x[k] = x[k – 1] + 4;**

 **}**

 **}**

**for(k = 4; k >= 0; k--)**

 **printf("%d ", x[k]);**

**6 10 6 8 4**

1. To assign values to all the elements of an array vect of size 5 in **reverse** order, the correct statements are:
2. for (i = 0; i < 5; ++i) scanf("%d",&vect[5]);
3. for (i = 5; i >= 0; ++i) scanf("%d",&vect[i]);
4. for (i = 4; i >= 0; --i) scanf("%d",&vect[i]);
5. for (i = 0; i <= 5; ++i) scanf("%d",&vect[5-i]);
6. Consider the following array:

 {8, 7, 5, 4, 5, 6, 4, 3}.

When calling linearSearch function (studied in class) using a target value (value to search for) of 5, the value returned by the function is

* 1. 3
	2. 5
	3. 2
	4. 4
1. Consider the following array:

 {8, 7, 5, 4, 5, 6, 4, 3}.

When calling linearSearch function (studied in class) using a target value (value to search for) of 10, the value returned by the function is

* 1. 0
	2. -1
	3. 8
	4. false
1. The correct prototype for a function receiving array A with values and copying them in array B in reverse order is:
2. void CopyArray(int A[], int \*B[], int n);
3. void CopyArray (int \*A[], int \*B[], int n);
4. void CopyArray (int \*A[], int B[], int n);
5. void CopyArray (int A[], int B[], int n);

#include <stdio.h> //6 points

int main() {

 int i, a[5] = {2, 1, 3, 2, 1};

for(i=1; i <= 5; i++)

 if(i<4)

 a[i] = a[i-1] + a[i+1];

 else

 a[i%5]=a[i-1]+a[(i+1)%5];

for(i=0;i<5;i++)

 printf("%d ",a[i]);

return 0;

 }

15 5 7 8 10

int A[6], i, n = 6;

for(i = 0; i < n; i++)

 if (i % 2 == 0)

 A[i] = 5 \* i;

 else

 A[n - i] = -5 \* i;

for(i = 0; i < n; i++)

 printf("%d\n", A[i]);

**0**

**-25**

**10**

**-15**

**20**

**-5**

|  |  |
| --- | --- |
|  | Consider the following array: |
|  | int x[7]={2, -1, 5, 3, 0, 4, 8};Consider a call to linear search function covered in class with array x and target value of 6. How many times is the target value 6 compared inside the function?  |
|  | A. | 5 times |
|  | B. | 7 times |
|  | C. | 8 times |
|  | D. | 6 times |

|  |
| --- |
| The proper prototype for a function that adds two arrays A and B of size **(**n**)** and put the result back in array A is: |
| A. | void addArray(int A[], int B[],int n); |
| B. | int A[] addArray(int A[], int B[], int n); |
| C. | void addArray(int A[], int B[], int \*A[], int n); |
| D. | void addArray(int A[n], int B[n], int n); |

|  |
| --- |
| What is the output after executing the code fragment shown below? |
| int A[6] = {8,11,12,19,24,33};int i, sum=0; for (i=0; i<6; i++){ if (A[i]%2 ==0 && A[i]%3==0) continue; printf("%d ", A[i]); sum += A[i];  if (sum>20)  break; } |
| A. | 8 11 12 19 24 33 |
| B. | 8 11 19 33 |
| C. | 8 11 19 |
| D. | 8 11 |

**The next five questions are based on the following incomplete code**

The code shown below is for the **main** and **modify** functions. **modify** is a logical function that receives an integer array named x of size m and it swaps each element in the array with another element from the same array having a random index. The random index will be generated by calling rand() function used in the lab. The **modify** function returns 0 if the array is not modified otherwise it returns 1.

#include<stdio.h>

#include<stdlib.h>

**\_\_\_\_\_Statement 1: Prototype\_\_\_\_\_\_;**

int main() {

 int x[8]={1,2,3,4,5,6,7,8};

 if(**\_Statement 2: Function Call\_**)

 printf("array modified");

 else

 printf("array not modified");

 return 0;

}

//below is the function definition of modify

**\_\_\_\_\_\_\_ Statement 1: Function Header\_\_\_\_\_\_**{

 int rand\_index, temp, i;

 /\* array x will not be modified if its size is 1\*/

 **\_\_\_\_\_Statement 3: Test Array Size\_\_\_\_**

 for(i = 0; i< m; i++){

 /\*Swap element at index i with element at random index \*/

 **\_\_\_\_\_Statement 4: generate random index\_\_\_\_\_**

 temp = x[rand\_index];

 **\_\_\_\_\_Statement 5: swap\_\_\_\_\_**

 x[i] = temp;

 }

 return 1;

}

|  |  |
| --- | --- |
|  | Statement 1 should be: |
|  | A. | int modify(int x[],int m) |
|  | B. | void modify (int x[],int m) |
|  | C. | void modify (int x,int m) |
|  | D. | int modify (int x[m]) |
|  | Statement 2 should be: |
|  | A. | modify(x,8) |
|  | B. | void modify(x[], 8) |
|  | C. | modify (x[], 8) |
|  | D. | int modify (x,8) |
|  | Statement 3 should be: |
|  | A. | if (m == 1) return 1; |
|  | B. | if (m = 1) return 0; |
|  | C. | if (m == 1) return 0; |
|  | D. | if (m = 1) return 1; |
|  | Statement 4 should be: |
|  | A. | rand\_index = rand()%(m+1); |
|  | B. | rand\_index = rand() % m-1; |
|  | C. | rand\_index = rand(); |
|  | D. | rand\_index = rand() % m; |
|  | Statement 5 should be: |
|  | A. | x[rand\_index] = temp; |
|  | B. | x[rand\_index] = x[i]; |
|  | C. | x[i]= x[rand\_index]; |
|  | D. | temp=x[i]; |

|  |  |
| --- | --- |
|  | Consider the following incomplete code fragment: |
|  | int x[3]={2,5,9};**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_//** call the function funprintf("%d %d %d\n", x[0],x[1],x[2]); The function fun is defined as follows:void fun (int \*a, int b, int \*c) { \*a = \*a + 2; b = b - 1; \*c = \*c + 1;}Which of the following is the correct call for the function fun to generate: 2 6 11 as the output? |
|  | A. | fun(&x[1], x[0], &x[2]); |
|  | B. | fun(&x[2], x[0], &x[1]); |
|  | C. | fun(&x[2], x[1], &x[0]); |
|  | D. | fun(x, x[0], x); |

|  |
| --- |
| What will be the output of the following code fragment? |
| int z[2] = {7, 9};ICS(z, z[1]);printf("%d %d", z[0], z[1]);where the function ICS is defined as:void ICS(int ar[],int r){ ar[0] = 2; r= 4;} |
| A. | 2 4 |
| B. | 7 4 |
| C. | 7 9 |
| D. | 2 9 |

#include <stdio.h>

int main(void)

{

 int a[7], j, k, b[] = {36,49,70};

 a[0] = 53;

 a[2] = 45;

 a[4] = 34;

 a[6] = 19;

 for(k=0; k<3; k++) {

 for (j=0; j<5; j=j+2) {

 if(b[k]>=a[j]) {

 a[j+1] = b[k];

 b[k] = a[j];

 break;

 }

 }

 }

 printf(“%d %d %d\n”, a[1], a[3], a[5]);

 printf(“%d %d %d\n”, b[0], b[1], b[2]);

 system(“pause”);

 return 0;

}

70 49 36

34 45 53

#include <stdio.h>

#define SIZE 10

int fun(int n);

int main(void)

{

 int a[SIZE] = {3,4,8,9,1,16,5,12,11,6};

 int i, c1 = 0, c2 = 0;

 for(i=0; i<SIZE; ++i) {

 if(fun(a[i]))

 c1++;

 else

 c2++;

 }

 printf(“%d\n”, c1);

 printf(“%d\n”, c2);

 system(“pause”);

 return 0;

}

int fun(int n)

{

 return n%4==0;

}

4

6

#include <stdio.h>

#define SIZE 5

void fun1(int a[], int size);

void fun2(int a[], int size);

int main(void)

{

 int x[SIZE] = {6,3,9,1,4};

 int i;

 fun1(x, SIZE);

 fun2(x, SIZE);

 system(“pause”);

 return 0;

}

void fun1(int a[], int size)

{

 int i;

 for(i=0; i<size; ++i) {

 if(i%2==1)

 a[i] += 2;

 else

 a[i] \*= 2;

 }

}

void fun2(int a[], int size)

{

 int i;

 for(i=0; i<size; ++i)

 printf(“%d\n”, a[i]);

}

12

5

18

3

8

#include <stdio.h> // **P2: 7 points**

#define SIZE 7

int main() {

 int a[SIZE],i;

 printf("Enter 7 integers\n");

 int count = 0;

 for(i = 1;i <= SIZE; i++){

 scanf("%d",&a[count%7]);

 count = count+3;

 }

 for(i=0;i < SIZE; i++)

 printf("%d\n",a[i]);

 return 0;

}

**Values typed by the user:**

**Enter 7 integers**

**5 4 6 3 2 7 1**

**57**

**3**

**4**

**1**

**2**

**6--------------------------------------**

include <stdio.h> // **P5: 9 points**

#define SIZE 6

void last(int \*x, int y, int n);

int main() {

 int a[SIZE],i;

 a[0]=2;a[2]=5;a[4]=8;

 for(i=1;i< SIZE;i=i+2)

 last(&a[i], a[i-1], i);

 for(i=4;i>=0;i=i-2)

 last(&a[i], a[i+1], i);

 for(i=0;i<SIZE;i++)

 printf("%d\n",a[i]);

 return 0;

}

void last (int \*x, int y, int n) {

 if( n%2 == 0)

 \*x = y+n;

 else

 \*x = y - n;

}

**1**

**1**

**4**

**2**

**7**

**3**--------------------------------------------------------------------------------------------------------------------------

#include<stdio.h> // **P6 6 points**

void check(int \*a,int \*b, int i){

 if(i%3==0)

 \*a=\*b+1;

 else if (i%2 == 0)

 \*a=\*b-1;

 else

 \*a = \*b;

 }

int main(void){

 int i,x[4]={2,3,7,8};

 int y[4];

 for (i = 0 ;i<4;i++ )

 check(&y[i],&x[3 - i],i);

 for (i = 0 ;i<4;i++ )

 printf("%d\n",y[i]);

 return 0;

}

**9**

**7**

**2**

**3**

#include<stdio.h>

int main(void){

 int n[8]={1, 1};

 int i;

 for(i=2; i<8; i++)

 n[i]=n[i-1]+n[i-2];

 for(i=0; i<8;i++)

 printf(“%d\n”, n[i]);

 return 0;

}

**1**

**1**

**2**

**3**

**5**

**8**

**13**

**21**

#include<stdio.h> **// 6 points**

void str (int a[],int \*b,int \*c,int \*d) {

a[5-\*b]=\*b;

 \*b=\*b+1;

a[5-\*c]=\*c;

\*c=\*c+1;

a[5-\*d]=\*d;

\*d=\*d+1;

}

int main(void){

int x[6]={0,1,2},i;

strange(x,&x[0],&x[1],&x[2]);

for(i=0;i<6;i++)

printf("%d\t",x[i]);

return 0;

 }

**1 2 3 2 1 0**

#include <stdio.h> **// 8 points**

void f1(int \*b);

void f2(int a, int b, int c, int d);

int main() {

int b[]={1,2,3,4},i;

for (i=0;i<4;i++)

 f1(&b[i]);

for (i=0;i<4;i++) {

 printf("%d\t",b[i]);

 b[i]=i;

}

printf("\n");

f2(b[0],b[1],b[2],b[3]);

for(i=0;i<4;i++)

 printf("%d\t",b[i]);

return 0;

}

void f1(int \*b) {

\*b=\*b\*2;

}

void f2(int a, int b, int c, int d) {

a+=1;

b+=2;

c+=3;

d+=4;

}

**2 4 6 8**

**0 1 2 3**