**Question 1**(1 point)

Which of the following statements is correct about pointers in C?

Question 1 options:

|  |  |
| --- | --- |
|  | A pointer can point to any type of variable. |
|  | A pointer is a constant that stores the address of a variable. |
|  | A pointer must always point to an int variable. |
|  | A pointer can only store string and integer values. |

**Question 2**(1 point)

What is the output of the following code?  
int x = 20;  
int \*ptr = &x;  
\*ptr = 30;  
printf(\%d\, x);

Question 2 options:

|  |  |
| --- | --- |
|  | 20 |
|  | 30 |
|  | 0 |
|  | Segmentation Fault |

**Question 3**(1 point)

What will the following code output?  
int arr[5] = {1, 2, 3, 4, 5};  
int \*p = arr;  
printf(\%d\, \*(p + 2));

Question 3 options:

|  |  |
| --- | --- |
|  | 1 |
|  | 2 |
|  | 3 |
|  | 4 |

**Question 4**(1 point)

What happens if you forget to free dynamically allocated memory?

Question 4 options:

|  |  |
| --- | --- |
|  | The program crashes. |
|  | The memory is automatically freed. |
|  | It results in a memory leak. |
|  | The memory becomes unusable. |

**Question 5**(1 point)

What will the following code output?  
int \*ptr = NULL;  
printf(\%d\, \*ptr);

Question 5 options:

|  |  |
| --- | --- |
|  | 0 |
|  | Undefined behavior |
|  | Segmentation Fault |
|  | Compiler Error |

**Question 6**(1 point)

*Saved*

What will the following code print?  
int \*p = malloc(sizeof(int));  
\*p = 100;  
free(p);  
printf(\%d\, \*p);

Question 6 options:

|  |  |
| --- | --- |
|  | 100 |
|  | 0 |
|  | Undefined behavior |
|  | Compilation error |

**Question 7**(1 point)

*Saved*

What will the following code print?  
int arr[3] = {10, 20, 30};  
int \*p = arr;  
printf(\%d\, \*(p+1) + \*(p+2));

Question 7 options:

|  |  |
| --- | --- |
|  | 10 |
|  | 50 |
|  | 30 |
|  | 60 |

**Question 8**(1 point)

*Saved*

Which of the following statements about pointers is correct?

Question 8 options:

|  |  |
| --- | --- |
|  | int \*p = NULL; is a valid way to initialize a pointer. |
|  | int p = NULL; correctly declares a pointer. |
|  | int \*p = &p; is valid and makes p point to itself. |
|  | Pointers can only point to variables of the same type as themselves. |

**Question 9**(1 point)

*Saved*

malloc() returns a pointer of type void\*.

Question 9 options:

|  |  |
| --- | --- |
|  | True |
|  | False |

**Question 10**(1 point)

*Saved*

In C, you can change the value of a constant integer using a pointer

Question 10 options:

|  |  |
| --- | --- |
|  | True |
|  | False |