

| 위치 | 오류유형 | 수정 전 | 수정 후 | | | | | | | | | | | | | | | | |
|--|--|--|---|----|----|---|-------------|---|---|----------------|---|---|------------------|--|------------------------------|--|----------------------|----------------------------------|------|
| 77p | 문제-본문 | <p style="text-align: right; font-size: small;">독학사 컴퓨터공학과 4단계_통합프로그래밍</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 5%;">번호</th> <th style="width: 25%;">표현</th> <th style="width: 70%;">설명</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>char a='A';</td> <td>char형 변수 a에는 문자 상수 A를 저장하기 위해 메모리 1개소리가 할당됨 <div style="border: 1px solid gray; padding: 2px; width: fit-content; margin: 5px 0;"> a </div> </td> </tr> <tr> <td>2</td> <td>char a[2]='A';</td> <td>char형 변수 a에는 문자 상수 A를 저장하기 위해 메모리 2개소리가 할당됨 <div style="border: 1px solid gray; padding: 2px; width: fit-content; margin: 5px 0;"> A </div> </td> </tr> <tr> <td>3</td> <td>char a[]="Blue";</td> <td>문자열이 7자리인 후 나머지 배열요소 a[4]-a[6]까지 공문자기 삽입됨 <div style="border: 1px solid gray; padding: 2px; width: fit-content; margin: 5px 0;"> B u e \0 \0 \0 </div> </td> </tr> </tbody> </table> <p>2 String 이용한 문자열</p> <p>String 키워드로 표현하며, 참조 자료형이다.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tbody> <tr> <td style="width: 50%;">String 변수명 = "표현하고자 하는 문자열";</td> <td style="width: 50%;">String 변수명 = new String("표현하고자 하는 문자열");</td> </tr> <tr> <td>String name = "홍길동";</td> <td>String name = new String("홍길동");</td> </tr> </tbody> </table> <p>String 객체는 문자열로 표현할 수도 있고, new 생성자를 이용하여 표현할 수도 있다. 직접 문자열로 표현하는 방식과 new 생성자를 이용하는 방식은 메모리 영역이 서로 상이하다.</p> <pre style="border: 1px solid gray; padding: 5px;"> public class StringExample { public static void main(String [] args) { String car1 = new String("Sonata"); //참조 자료형, heap 영역 생성 String car2 = "grandeur"; //Constant Pool 영역 String car3 = "santafe"; //Constant Pool 영역 } } </pre> <p>* 생성되는 객체 수는 heap 영역에 1개, Constant Pool 영역에 1개 총 2개 생성됨</p> <p>컴파일될 때 문자열 상수는 String 객체를 생성하게 된다.</p> <p style="text-align: right; font-size: small;">제1장 자료형 77</p> | 번호 | 표현 | 설명 | 1 | char a='A'; | char형 변수 a에는 문자 상수 A를 저장하기 위해 메모리 1개소리가 할당됨 <div style="border: 1px solid gray; padding: 2px; width: fit-content; margin: 5px 0;"> a </div> | 2 | char a[2]='A'; | char형 변수 a에는 문자 상수 A를 저장하기 위해 메모리 2개소리가 할당됨 <div style="border: 1px solid gray; padding: 2px; width: fit-content; margin: 5px 0;"> A </div> | 3 | char a[]="Blue"; | 문자열이 7자리인 후 나머지 배열요소 a[4]-a[6]까지 공문자기 삽입됨 <div style="border: 1px solid gray; padding: 2px; width: fit-content; margin: 5px 0;"> B u e \0 \0 \0 </div> | String 변수명 = "표현하고자 하는 문자열"; | String 변수명 = new String("표현하고자 하는 문자열"); | String name = "홍길동"; | String name = new String("홍길동"); | a[1] |
| 번호 | 표현 | 설명 | | | | | | | | | | | | | | | | | |
| 1 | char a='A'; | char형 변수 a에는 문자 상수 A를 저장하기 위해 메모리 1개소리가 할당됨 <div style="border: 1px solid gray; padding: 2px; width: fit-content; margin: 5px 0;"> a </div> | | | | | | | | | | | | | | | | | |
| 2 | char a[2]='A'; | char형 변수 a에는 문자 상수 A를 저장하기 위해 메모리 2개소리가 할당됨 <div style="border: 1px solid gray; padding: 2px; width: fit-content; margin: 5px 0;"> A </div> | | | | | | | | | | | | | | | | | |
| 3 | char a[]="Blue"; | 문자열이 7자리인 후 나머지 배열요소 a[4]-a[6]까지 공문자기 삽입됨 <div style="border: 1px solid gray; padding: 2px; width: fit-content; margin: 5px 0;"> B u e \0 \0 \0 </div> | | | | | | | | | | | | | | | | | |
| String 변수명 = "표현하고자 하는 문자열"; | String 변수명 = new String("표현하고자 하는 문자열"); | | | | | | | | | | | | | | | | | | |
| String name = "홍길동"; | String name = new String("홍길동"); | | | | | | | | | | | | | | | | | | |
| | | 수정 사유 | 오타 수정 | | | | | | | | | | | | | | | | |
| 118p [switch 문장 및 도식 예시] 그림 및 설명 수정 | 개념, 공식-설명 | <p>break가 없는 경우 그 다음 case를 실행하게 되며 default는 무조건 실행한다.</p> | <p>break가 없는 경우에는 case와 일치하는 문장을 포함해서 이후 default문까지 실행된다. 반면 break가 있는 경우에는 case와 일치하는 문장만 실행되며, default문은 switch문에서 case에 일치하는 값이 없을 때 실행된다.</p> | | | | | | | | | | | | | | | | |
| | | 수정 사유 | 설명 보완 | | | | | | | | | | | | | | | | |

| 위치 | 오류유형 | 수정 전 | 수정 후 | | | | | | |
|---|--|---|---|--|---|--|------------------------------|-----------------------------------|-----|
| 123p | 문제-본문 | <p style="text-align: right; font-size: small;">독학사 컴퓨터공학과 4단계_통합프로그래밍</p> <p>(1) for문과 while문 비교 for문과 while문을 선택해야 할 때, 반복 대상이 정해져있다면 while문의 사용을 권장한다.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 50%;">for문</th> <th style="width: 50%;">while문</th> </tr> </thead> <tbody> <tr> <td> <pre>for(i = 0; i <= 10; i++) { printf("%d", i); }</pre> </td> <td> <pre>i = 0; //초기화 수행 while(i <= 10) { //조건부의 printf("%d", i); i++; //증감처리 }</pre> </td> </tr> <tr> <td> <pre>for(i) { //무한루프 }</pre> </td> <td> <pre>while(true) { //무한루프 }</pre> </td> </tr> </tbody> </table> <p>for문은 초기화, 조건식, 증감식을 한곳에 모아놓은 형태이고 while문은 조건식만 있다. while문을 사용할 때는 무한루프에 빠지지 않도록 조심해야 한다.</p> <p>(2) while문은 조건식 생략 불가 for문과 달리 while문은 조건식을 생략할 경우 오류가 발생하기 때문에 조건식은 필수이다.</p> <pre>while() { //오류임, 조건식 없음 }</pre> <p>(3) while문을 이용한 자연수 합 연산</p> <pre>1 #include <stdio.h> 2 int main() { 3 int n, i, sum = 0; 4 printf("종수 입력: "); 5 scanf("%d", &n); //i의 값 입력 6 i = 1; //i의 값 초기화 7 8 while(i <= n) { 9 sum += i; //sum = i와 동일 10 i++; //i의 값을 1씩 증가 11 } 12 printf("Sum = %d\n", sum); 13 return 0; 14 }</pre> <p>while문을 사용할 때는 조건식을 통해 while문을 빠져나갈 수 있도록 처리해야 한다.</p> <p style="text-align: right; font-size: small;">제3장 제어문 123</p> | for문 | while문 | <pre>for(i = 0; i <= 10; i++) { printf("%d", i); }</pre> | <pre>i = 0; //초기화 수행 while(i <= 10) { //조건부의 printf("%d", i); i++; //증감처리 }</pre> | <pre>for(i) { //무한루프 }</pre> | <pre>while(true) { //무한루프 }</pre> | ++i |
| for문 | while문 | | | | | | | | |
| <pre>for(i = 0; i <= 10; i++) { printf("%d", i); }</pre> | <pre>i = 0; //초기화 수행 while(i <= 10) { //조건부의 printf("%d", i); i++; //증감처리 }</pre> | | | | | | | | |
| <pre>for(i) { //무한루프 }</pre> | <pre>while(true) { //무한루프 }</pre> | | | | | | | | |
| | | 수정 사유 | 오타 수정 | | | | | | |
| 125p | 문제-본문 | <p style="text-align: right; font-size: small;">독학사 컴퓨터공학과 4단계_통합프로그래밍</p> <p>(2) do-while 예시 소스를 while문으로 작성</p> <pre>#include <stdio.h> int main() { int i = 0; //i 초기화 //do에 해당하는 부분 printf("do-while %d\n", i); //처음 한 번은 실행 i++; //처음 한 번은 실행 while(i < 100) //i가 100보다 작을 때 반복, 0부터 99까지 증가하면서 100번 반복 { printf("do-while %d\n", i); //do-while의 i의 값을 함께 출력 i++; //i를 1씩 증가시킴 } return 0; }</pre> <p>프로그램 작성 시 대부분 do-while문보다는 while문을 통해 작성한다.</p> <p>4) foreach문 인자로 들어온 iterable-item 내부 인덱스 값까지 알아서 순환을 해주는 반복문으로, 일반적인 for 반복문과 동일하게 for문과 달리 반복문 내에 카운터 변수를 선언하고 출력() 다음 배열 이름을 순서대로 선언한다. 일반적으로 배열이나 Collection 클래스(Array, List ... 등)를 반복하는 데 사용한다.</p> <p>(1) for문과 비교한 foreach문법</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tbody> <tr> <td style="width: 50%;"> <pre>int[] arr = {0, 1, 2, 3, 4}; for (int i = 0; i < arr.length; i++) { System.out.println(arr[i]); //0 1 2 3 4 }</pre> </td> <td style="width: 50%;"> <pre>int[] arr = {0, 1, 2, 3, 4}; for (int i : arr) { System.out.println(i); //0 1 2 3 4 }</pre> </td> </tr> </tbody> </table> <p>foreach 반복문을 사용함으로써 간편하는 복잡한 배열이나 리스트의 크기를 구할 필요가 없다. 이중 for문이나 복잡한 반복문에 적합하며, 인덱스를 생성해 접근하는 대신 for문보다 수행속도가 조금 더 빠르다.</p> <p style="text-align: right; font-size: small;">제3장 제어문 125</p> | <pre>int[] arr = {0, 1, 2, 3, 4}; for (int i = 0; i < arr.length; i++) { System.out.println(arr[i]); //0 1 2 3 4 }</pre> | <pre>int[] arr = {0, 1, 2, 3, 4}; for (int i : arr) { System.out.println(i); //0 1 2 3 4 }</pre> | iterable-item | | | | |
| <pre>int[] arr = {0, 1, 2, 3, 4}; for (int i = 0; i < arr.length; i++) { System.out.println(arr[i]); //0 1 2 3 4 }</pre> | <pre>int[] arr = {0, 1, 2, 3, 4}; for (int i : arr) { System.out.println(i); //0 1 2 3 4 }</pre> | | | | | | | | |
| | | 수정 사유 | 오타 수정 | | | | | | |

| 위치 | 오류유형 | 수정 전 | 수정 후 |
|-----------------|-------|--|--|
| 147p 번호 : 10 | 정답 | <p>● 10번 문제 제시문 첫 번째 줄 int score = 98</p> <p>● 10번 문제 정답 ①</p> <p>● 10번 문제 해설 break문을 통해 case문을 빠져나가고 default는 무조건 처리가 된다.</p> | <p>● 10번 문제 제시문 첫 번째 줄 수정 (: 추가) int score = 98;</p> <p>● 10번 문제 정답 수정 ②</p> <p>● 10번 문제 해설 수정 score 값은 9가 되며, 조건에 맞는 case문인 A가 출력되고 break문을 통해 종료된다. 참고로 4행이 case 10: printf("Z"); 이라고 가정할 경우 break가 없어도 조건에 맞지 않아 case 10은 실행이 되지 않는다.</p> |
| | | 수정 사유 | 10번 문제 제시문, 정답, 해설 수정 |
| 173p | 문제-본문 | <p style="text-align: right; font-size: small;">독학사 컴퓨터공학과 4단계_통합프로그래밍</p> <div style="border: 1px solid gray; padding: 5px; margin-bottom: 10px;"> <p>this 키워드의 제약 조건</p> <ul style="list-style-type: none"> • this 키워드는 클래스의 멤버함수에서만 사용할 수 있다. • 멤버함수라도 정적 멤버함수는 this를 사용할 수 없다. </div> <p>멤버가 아닌 함수는 어떤 객체에도 속하지 않기 때문에 this 키워드는 클래스의 멤버함수에서만 사용이 가능하며, 정적 멤버함수는 객체가 생성되기 전에 호출될 수 있으므로, this를 사용할 수 없다.</p> <p style="text-align: center; font-size: x-small;">[C++ this 키워드의 사용 예시]</p> <pre style="font-family: monospace; font-size: x-small;"> 1 class TestClass { 2 3 int num; //멤버변수 선언 4 5 public: 6 Test(int num) { 7 this->num = num; //멤버변수의 객체변수를 구분하기 위해 this 사용 8 } 9 }; </pre> <p>위의 예시에서 this -> num은 멤버변수 num을 의미한다. 즉, 생성자에서의 객체변수 num을 멤버변수 num으로 초기화하는 코드이다. this 키워드를 사용하는 다른 예시를 살펴보자.</p> <p style="text-align: center; font-size: x-small;">[C++ this 키워드의 사용 예시]</p> <pre style="font-family: monospace; font-size: x-small;"> 1 class TestClass { 2 vector<int> vec; //멤버변수 선언 3 int num; //멤버변수 선언 4 5 public: 6 Test& pushBackNumber(int num) { 7 vec.push_back(num); 8 return *this; //자기 자신의 의미로 this 사용 9 } 10 }; 11 12 int main() { 13 Test t1; 14 t1.pushBackNumber(5).pushBackNumber(6).pushBackNumber(7); 15 return 0; 16 } </pre> <p style="text-align: right; font-size: x-small;">제정 클래스 173</p> | <p>● 첫 번째 체크 부분 TestClass(int num)</p> <p>● 두 번째 체크 부분 TestClass& pushBackNumber(int num)</p> <p>● 세 번째 체크 부분 TestClass t1</p> |
| | | 수정 사유 | 내용 수정 |

도서의 오류로 학습에 불편드린 점 진심으로 사과드립니다.
더 나은 도서를 만들기 위해 노력하는 시대교육그룹이 되겠습니다.