POSTECH 컴퓨터공학과 부정행위 정의

POSTECH 컴퓨터공학과 학부위원회

2007년 8월

제 1 절 부정행위 정의의 취지

본 문서는 대학생활에서 일어날 수 있는 부정행위의 의미를 정의하고 예를 제시하여 학생들의 부정행위에 대한 이해를 돕고자 작성하였다. 부정행위에 대한 이해가 부족한 학생들은 자신이 투자한 시간이 얼마인지만 중요시하고 그 과정은 고려하지 않은채 자신이 작업한 결과가 부정행위에 속하는 지를 알지 못하는 경우가 많다. 본 문서는 부정행위의 정확한 의미를 제시하여 부정행위를 미연에 방지함으로써 건전한 분위기에서 서로 경쟁하는 진취적인 POSTECH 컴퓨터공학과 학풍 정립을 고취시키고자 함에 그 취지가 있다.

제 2 절 처벌 대상 부정행위

부정행위는 크게 강의실에서 일어나는 것과 강의실 밖에서 일어나는 것으로 나뉜다. 강의실에서 일어나는 부정행위는 대리출석, 퀴즈 및 시험에서의 치팅(cheating) 등이 있다. 강의실 밖에서 일어나는 부정행위는 숙제나 보고서 작성시 표절(plagiarism), 허락되지 않은 협력행위(collaboration) 등이 있다.

대리출석

다른 학생이 자신의 출석을 대신 완료해 주는 모든 행위를 말한다.

퀴즈 및 시험에서의 치팅

퀴즈 및 시험 시간에 허락된 자료 외의 다른 자료를 참고하거나 다른 학생의 도움을 받아서 문제를 해결하는 행위를 말한다. 즉 허락된 방식 외의 어떤 수단을 이용하여 문제 해결하는 모든 행위를 지칭한다. 어떤 자료도 참고할 수 없는 시험에서 미리 준비해간 자료를 이용하는 경우가 한 예이다. 다른 예로는 시험 시간에 메모지, 전화, 계산기, 또는 직접 대화 등을 이용하여 다른 학생에게 자료나 아이디어를 전달하는 경우가 있다.

표절

표절행위는 궁극적으로 지식인으로서 학생의 정직성과 관련된 문제이다. 표절의 기본적인 의미는 다른 사람이 작성한 자료를 인용없이 그대로 혹은 변형하여 쓰는 것을 말한다. 여기서 말하는 자료는 이미 출판된 자료 뿐만이 아니라 출판되지는 않았지만 출처가 자신이 될 수 없는 모든 자료를 통칭한다. 출판된 자료는 책이나 논문 등과 같이 물리적인 증거가 있는 것과 인터넷 등에서 찾을 수 있는 것들을 모두 포함한다. 출처가 자신이 아닌 자료는 다른 사람이 작성한 문제의 해결책, 다른 저자가 정리한 아이디어, 타인과의 대화에서 얻은 자료 등이 포함된다. **즉 표절은 인용없이도 사용이 허락되는 자료(예를 들면 교과서, 강의 노트)외에 학생이 제출하는 자료가 자신의 고유한 작업물인지 아닌지에 따라서 결정되는 부정행위이다.** 예를 들어서 어떤 숙제의 경

우 교과서만 참고할 수 있다고 했다면, 학생이 제출하는 자료 중에서 교과서에 나오지 않는 모든 것은 학생의 고유한 작업물이거나 그 출처가 명시된 자료이어야만 한다.

어떤 과목은 과목의 특성상 다른 사람의 자료를 찾아서 쓰는 것이 허락되는 경우가 있다. 다른 사람의 자료를 쓰는 것이 허락되지 않은 경우에는, 학생이 제출하는 모든 자료는 학생 스스로가 만들어낸 것이어야 하며 그렇지 않은 경우에는 표절로 간주된다. 다른 사람의 자료를 참고하는 것이 허락되는 경우에도, 학생이 제출하는 자료 중 스스로 창안해 낸 것과 다른 사람의 아이디어를 빌려 온 것을 뚜렷하게 구별해야 한다. 이를 구별하지 않고 자료를 만들어 제출하는 것도 역시 표절로 간주된다.

표절은 그 판정의 기준이 핵심 아이디어가 어떻게 전달되었는지이며 구체적으로 어떻게 아이디어가 표현되었는지 여부가 아니다. 따라서 직접적으로 복사하는 경우 뿐만 아니라, 같은 내용의 다른 문장으로 바꿔쓰기(paraphrase) 또는 내용 요약 등의 경우에도 적절한 인용이 없다면 표절로 간주된다.

인용을 할 때에는 반드시 **관련된 부분 직후에** 출처를 적어야 한다. 문서의 마지막에 참조한 문헌 등을 나열하는 것은 자신이 작성한 자료의 어느 부분이 어떤 문헌을 참고했는지를 알려주지 않으므로 전혀 의미가 없다.

컴퓨터 프로그램의 경우 자신의 힘으로 처음부터 끝까지 작성한 경우 외 모든 경우가 표절로 간주된다. 따라서 이미 작성된 프로그램을 수정한 경우는 표절로 간주된다. 이는 최종 프로그램이 처음 참고한 프로그램과 완전히 달라 보이는 경우도 마찬가지이다. 또한 문제 해결과 직접 상관이 없어 보이는 테스트 코드 및 입출력 코드 등의 경우에도 동일한 판정 기준을 적용 받는다. 프로그램의 표절 판정에는 표절로 인정되는 부분이 전체 프로그램에서 어느 정도의 비중을 차지하는지는 중요하지 않다.

표절은 학생이 의도적이든 또는 단순한 부주위로 인용을 하지 않은 경우를 구별하지 않는다. 즉 인용하는 것을 실수로 잊고 자료를 제출한 경우에도 기본적으로 표절행위로 간주된다. 따라서 학생이 다른 출처를 찾아서 자료 수집을 시작할 때부터 이미 자신의 모든 행동에 대해서 책임을 져야한다. 실수로 인용이 불충분한 상태에서 제출된 자료가 나중에 표절행위로 인정되면 학생은 항의를 할 수 없다. 학생들은 이 점을 분명하게 이해하고 염두해 두어야 한다.

새로운 자료를 작성할 때 인용의 필요성이 확실하지 않은 경우가 흔히 발생한다. 이런 경우 가능하면 불필요해 보이는 인용도 포함을 시키는 것이 더 나은 선택이다. 왜냐하면 인용을 하지 않는 것은 처벌을 받을 수 있는 잘못된 행위이지만, 불필요해 보이는 인용이라도 포함시키는 것은 비록 작성된 자료를 조금 읽기 힘들게 만들 가능성은 있지만 적어도 처벌로 이어지는 행위는 아니기 때문이다.

표절의 경우 **자료를 제공한 사람(copyee)과 자료를 도용한 사람(copier)을 구별하지 않는다.** 이는 다른 학생으로부터 자료를 부탁받은 학생이 부담없이 거절할 수 있는 이유를 제공할 수 있도록 하여 표절을 미연에 방지하도록 하기 위함이다. 자료를 제공한 사람은 의도적인 표절에 관련되지 않고 도용한 사람이 모든 것을 진행한 경우에도 동일하게 처벌을 받게 된다. **따라서 학생들은 자신이 작업한 결과를 다른 학생들이 접근하지 못하게 하도록 할 책임이 있다.**

협력행위

부정행위로서의 협력행위는 다른 학생과의 토론이나 자료의 교환이 허락되지 않는 상황에서 이를 어기는 경우를 말한다. 많은 경우 협력행위의 결과는 표절의 형태로 나타나지만 그렇지 않은 경우도 있다. 협력행위가 금지된 숙제 등의 경우에도 학생들은 우리 사회의 정서상 단순한 형태의 질문 및 답변, 정보 교환 등은 피할 수 없다고 생각할 수 있으나 이는 공정한 학생 평가를 위해서는 인정될 수 없다. 왜냐하면 모든 학생들이 정보를 교환하거나 질문 및 답변을 할 수 있는 급우가 있는 것은 아니기 때문이다. 더구나 많은 경우 두명이 아닌 여러명의 학생들이 협력하게 되는데, 이는 혼자서 숙제를 해야하는 학생들의 사기를 심각하게 저하시킨다.

표절과 마찬가지로 협력행위의 허락 기준은 각 과목의 특성을 고려하여 과목의 책임자가 결정한다. 협력행위가 허락되지 않는 경우에는 학생들이 어떤 형태로든 아이디어를 공유하면 안되며, 아이디어가 공유되어 결과물로 제출되는 경우는 표절로 간주된다. 협력행위가 허락되는

경우에도 학생들은 자신이 혼자서 창안하지 않은 모든 자료의 출처를 함께 협력한 학생들의 이름 형식으로 반드시 명시해야 한다. 자신이 혼자서 창안하지 않은 자료는 다른 학생과의 토론을 통해서 얻은 아이디어 외에도 수치 데이터, 그림 정보 등 모든 매체를 포함한다. 협력행위가 허락된 경우에도 학생들이 인용을 제대로 하지 않은 경우는 표절로 간주되며 이는 전적으로 학생의 책임이다.

각 과목을 담당하는 교수는 협력행위가 허락되는지 아닌지를 강의계획서나 각 숙제 자료에 명시할 의무가 있다. **따로 명시하지 않는 경우는 협력행위는 허락되지 않는 것으로 해석된다.**

협력행위는 결과물이 표절의 범주에 속하는 여부에 상관없다. 즉 협력행위와 표절은 별도의 부정행위로 간주된다. 또한 협력행위는 실제 참여 비중에 관계없이 **연루된 모든 학생을 모두 동일하게 참여한 것으로 간주한다.**

제 3 절 표절의 예

3.1 C 프로그램 표절의 예 (http://www.princeton.edu/pr/pub/integrity/pages/plagiarism.html 참조)

```
quicksort (int a[ ], int 1, int r)
    {
        int v, i, j, t;
        if (r > 1)
            {
                v = a[r]; i = 1-1; j = r;
                for (;;)
                    {
                        while (a[++i] < v) ;
                        while (a[--j] > v) ;
                        if (i >= j) break;
                        t = a[i]; a[i] = a[r]; a[r] = t;
                    }
                 t = a[i]; a[i] = a[r]; a[r] = t
                 quicksort (a, l, i-l);
                 quicksort (a, i+l, r);
            }
    }
```

위 프로그램은 원본 프록그램으로서 quick sort를 구현하는 C 함수이며 Robert Sedgewick의 Algorithms in C (Addison Wesley, New York, 1990) 118쪽에 있다.

```
mysort (int data[ ], int x, int y) {
    int pivot;
    int i, j;
    int temp;
    if (y > x) {
        pivot = data[y]; i = x-1; j = r;
        while (1) {
            while (data [++i] < pivot) ;
            while (data [—j] > pivot) ;
            if (i >= j) break;
            temp = data [i]; data [i] = data [y]; data [y] = temp;
        }
        temp = data [i]; data [i] = data [y]; data [y] = temp;
        mysort (data, x, i-1);
        mysort (data, i+1, y);
    }
}
```

위 프로그램은 표절로 판정되는 예이다. 위 프로그램은 원본 프로그램과 비교했을 때 정확하게 똑같은 구조를 가지고 있다. 프로그램의 외관상 원본 프로그램과 완전히 달라보이지만 두 프로그램의 의미는 사실 동일하다. 원본 프로그램과의 차이점은 변수명 및 함수 이름이 달라졌고 `for(;;)` 구문을 `while(1)` 구문으로 바꾸었으며 띄워쓰기가 달라진 것 뿐이다.

```
#define Swap(A,B) { temp=(A); (A)=(B); (B)=(A); }
void mysort (const int * data, int x, int y){
    int temp;
    while (y > x){
        int pivot = data[y];
        int i = x-1;
        int j = r;
        while (1){
            while (data [++i] < pivot) { /*do nothing*/ }
            while (data [--j] > pivot) { /*do nothing*/ }
            if (i >= j) break;
            swap (data [i], data [y];
        }
        swap (data [i], data [j];
        mysort (data, x, i-1);
        x = i+1;
    }
}
```

위의 프로그램도 표절로 판정되는 예이다. 매크로를 쓰는 등의 변화가 주어졌지만 원본 프로그램에서 변형된 형태이기 때문이다.

3.2 SML 프로그램 표절의 예 (CSE-321 Programming Languages 2006 봄학기)

```
(*
 *  typingDown : context -> Cml.exp -> Cml.tp
 *)
and typingDown cxt (Cml.Var x) = cxt x
 |  typingDown cxt (Cml.App(e, i)) =
      if isElim e andalso isIntro i then
        case (typingDown cxt e) of
          Cml.Fun(A, B) => if typingUp cxt i A then B else raise TypeError
        | _             => raise TypeError
      else
        raise TypeError
 |  typingDown cxt (Cml.Fst e) =
      if isElim e then
        case (typingDown cxt e) of
          Cml.Prod(A1, A2)  => A1
        | _                 => raise TypeError
      else
        raise TypeError
 |  typingDown cxt (Cml.Snd e) =
      if isElim e then
        case (typingDown cxt e) of
          Cml.Prod(A1, A2)  => A2
        | _                 => raise TypeError
      else
        raise TypeError
 |  typingDown cxt (Cml.Withtype (i, A)) =
      if isIntro i then
        if typingUp cxt i A then
          A
        else
          raise TypeError
      else
        raise TypeError
(* typingDown: context -> Cml.exp -> Cml.tp
 *
 * desc : Return a type which hold context |- E downarrow A
 *)
and typingDown cxt exp =
  case exp of
    Cml.Var x => cxt x
  | Cml.App(e, I) =>
      if isElimExp e andalso isIntroExp I then
        case (typingDown cxt e) of
          Cml.Fun(A, B) => if typingUp cxt I A then B else raise TypeError
        | _             => raise TypeError
      else
        raise TypeError

  | Cml.Fst e =>
      if isElimExp e then
        case (typingDown cxt e) of
          Cml.Prod(A1, A2) => A1
        |                  _ => raise TypeError
      else
        raise TypeError

  | Cml.Snd e =>
      if isElimExp e then
        case (typingDown cxt e) of
          Cml.Prod(A1, A2) => A2
        |                  _ => raise TypeError
      else
        raise TypeError

  | Cml.Withtype(I, A) =>
      if isIntroExp I then
        if typingUp cxt I A then A
        else
          raise TypeError
      else
        raise TypeError
```

위의 두 프로그램은 서로 표절로 판정된 경우이다. 오른쪽 프로그램은 왼쪽 프로그램의 `typingDown` 함수의 두번째 인자를 `case` 구문으로 분석하는 것과 띄어쓰기에 약간 변형을 준 것, 그리고 몇 몇 변수의 이름을 바꾼 것 외에는 아무 아무 차이가 없다.

3.3 증명 표절의 예 (CSE-321 Programming Languages 2006 봄학기)

Theorem 2 (Substitution). *If* $\Gamma \vdash e : A$ *and* $\Gamma, x : A \vdash e' : C$, *then* $\Gamma \vdash [e/x]e' : C$.

Proof. By rule induction on the judgment $\Gamma' = \Gamma, x : A \vdash e' : C$.

Case $\dfrac{y : C \in \Gamma'}{\Gamma' \vdash y : C}$ Var where $e' = y$:

if $y = x$ then, $[e/x]y = e$	*by* $y = x$
$\Gamma' \vdash [e/x]y : A = C$	*by* $y : C = x : A$
else $\Gamma' \vdash [e/x]y = y : C$	*by* $y \neq x$

Case $\dfrac{\Gamma', y : A' \vdash e2 : B}{\Gamma' \vdash \lambda y{:}A'.\, e2 : A' \to B} \to\mathsf{I}$ where $e1 = \lambda y : A'.\, e2$:

if $x \neq y$ then, $\Gamma', y : A', x : A \vdash e2 : B$	*by Unordered set on the subderivation*
$\Gamma', y : A' \vdash e : A$	*by weakening on the* $\Gamma \vdash x : A$
$\Gamma', y : A' \vdash [e/x]e2 : B$	*induction Hypothesis*
$\Gamma' \vdash \lambda y{:}A'.\, [e/x]e2 : A' \to B$	*by the rule* $\to\mathsf{I}$
$[e/x]e1 = \lambda y{:}A'.\, [e/x]e2$	*by the definition of substitution*
else $[e/x]e1 = e'$	*by the definition of subsitution*

Theorem 2 (Substitution). *If* $\Gamma \vdash e : A$ *and* $\Gamma, x : A \vdash e' : C$, *then* $\Gamma \vdash [e/x]e' : C$.

Proof. By rule induction on the judgment $\Gamma' = \Gamma, x : A \vdash e' : C$.

Case $\dfrac{y : C \in \Gamma'}{\Gamma' \vdash y : C}$ Var where $e' = y$:

$y = x$	*Subcase 1*
$[e/x]y = e$	*by* $y = x$
$\Gamma' \vdash [e/x]y : A = C$	*by* $y : C = x : A$
$y \neq x$	*Subcase 2*
$\Gamma' \vdash [e/x]y = y : C$	*by* $y \neq x$

Case $\dfrac{\Gamma', y : A' \vdash e'' : B}{\Gamma' \vdash \lambda y{:}A'.\, e'' : A' \to B} \to\mathsf{I}$ where $e' = \lambda y{:}A'.\, e''$:

$x \neq y$	*Subcase 1*
$\Gamma', y : A', x : A \vdash e'' : B$	*by Unordered set on the subderivation*
$\Gamma', y : A' \vdash e : A$	*by weakening on the* $\Gamma \vdash e : A$
$\Gamma', y : A' \vdash [e/x]e'' : B$	*induction Hypothesis*
$\Gamma' \vdash \lambda y{:}A'.\, [e/x]e'' : A' \to B$	*by the rule* $\to\mathsf{I}$
$[e/x]e' = \lambda y{:}A'.\, [e/x]e''$	*by the definition of substitution*

위의 두 증명은 표절로 판정된 경우이다. 두 증명은 변수사용에 있어서 약간의 변화가 주어졌지만 그 의미상 완전히 같다. 또한 특정 단어의 첫자를 대문자로 쓴 것 또한 표절에서만 일어날 수 있는 실수이다. (예: Unordered, Hypothesis)

3.4 C 프로그램 표절의 예 (CSE-101 전자계산입문 2007 봄학기)

```
int find2(FILE *fp, char value[], char to[]){ // fp파일 포인터에서 두번째 column의 값 중 value배열에 저장된 값과 일치하Â 경우, 첫번째 column의 값을 to배열에 저장한다. 일치하는 값이 없으면 0 리턴.
  char gar, cmp[50];
  while(fscanf(fp, "%s\t%s", to, cmp)!=EOF){
    if(strcmp(cmp, value)==0)
      return 1;
  }
  return 0;
}
int find_2(FILE *fp, char f_value[], char s_value[]){ // fp파일 포인터에서 f_value와 s_value의 값이 각각 첫번째 column과 두번째 column에서 일치하는 경우 1, 그런 경우가 없다면 0을 리턴한다
  char comp[50];
  while(find(fp, f_value)!=0){
    fscanf(fp, "\t%s", comp);
    if(strcmp(comp, s_value)==0)
      return 1;
  }
  return 0;
}

int find_grade_stu(char stu_num[]){ //regists.txt파일에서 해당되는 학번을 가진 사람이 신청한 과목의 총 학점을 리턴해준다.
  FILE *file1, *file2;
  int grade=0, grade_plus;
  char stu_cmp[20], subject[20], gar1[20], gar2[20];
  file1=fopen("classes.txt", "r");
  file2=fopen("regists.txt", "r");
  while(fscanf(file2, "%s\t%s", stu_cmp, subject)!=EOF){
    if(strcmp(stu_cmp, stu_num)==0){
      rewind(file1);
      find(file1, subject);
      fscanf(file1, "%s", gar1);
      fscanf(file1, "%s", gar2);
      grade+=gar2[4]-48;
    }
  }
  fclose(file1);
  fclose(file2);
  return grade;
}
```

```
int check2(FILE *file, char val1[], char val2[])  /*파일 내부의 문서와 입력값 비교 프로그램2*/
{
        char com[50];
        while(fscanf(file, "%s\t%s", val2, com)!=EOF)
        {
                if(strcmp(com, val1)==0)
                        return 1;
        }
        return 0;
}

int match(FILE *file, char *val1[], char *val2[])  /*파일 내부의 문서와 입력값 비교 프로그램3*/
{
        char con[100];
        while(check1(file, val1)!=0)
        {
                fscanf(file, "\t%s", con);

                if(strcmp(con, val2)==0)
                {
                        return 1;
                }
        }
        return 0;
}

int calcul_max_grade(char *stid[])  /*학생이 신청한 학점 계산 프로그램*/
{
        int sumgra=0;
        char stunum[20], subject[20], gar1[20], gar2[20];
        FILE *clas, *regi;

        clas=fopen("classes.txt", "r");
        regi=fopen("regists.txt", "r");

        while(fscanf(regi, "%s\t%s", stunum, subject)!=EOF)
        {
                if(strcmp(stunum, stid)==0)
                {
                        rewind(clas);
                        check1(clas, subject);
                        fscanf(clas, "%s", gar1);
                        fscanf(clas, "%s", gar2);
                        sumgra+=gar2[4]-48;
                }
        }
        fclose(clas);
        fclose(regi);
        return sumgra;
}
```

위의 두 프로그램은 서로 표절로 판정된 경우이다. 두 프로그램은 변수명의 선택에 차이가 있지만 정확하게 같은 구조를 가지고 있다.